



[illegible]



DLDRIVER  
Table of contents

- VAX/VMS RL11/RL01,RL02 DISK DRIVER<sup>6 5</sup>

16-SEP-1984 00:17:29 VAX/VMS Macro V04-00

Page 0

(1)	113	EXTERNAL AND LOCAL DEFINITIONS
(1)	299	STANDARD TABLES
(1)	522	CONTROLLER INITIALIZATION ROUTINE
(1)	572	UNIT INITIALIZATION ROUTINE
(1)	695	DRIVER SPECIFIC SUBROUTINES
(1)	718	FDT ROUTINE - TEST TRANSFER BYTE COUNT ALIGNMENT
(1)	754	START I/O ROUTINE
(1)	1532	INTERRUPT SERVICE ROUTINE
(1)	1593	REGISTER DUMP ROUTINE
(1)	1632	MOVE TO USER BUFFER ROUTINE
(1)	1676	MOVE FROM USER BUFFER ROUTINE

```
0000 1      .TITLE DLDRIVER - VAX/VMS RL11/RL01,RL02 DISK DRIVER
0000 2      .IDENT 'V04-000'
0000 3
0000 4      *****
0000 5      *
0000 6      * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 7      * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 8      * ALL RIGHTS RESERVED.
0000 9      *
0000 10     * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 11     * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 12     * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 13     * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 14     * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 15     * TRANSFERRED.
0000 16     *
0000 17     * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 18     * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 19     * CORPORATION.
0000 20     *
0000 21     * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 22     * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 23     *
0000 24     *
0000 25     *****
0000 26
0000 27
0000 28     FACILITY:
0000 29
0000 30     VAX/VMS RL11/RL01,RL02 DISK DRIVER
0000 31
0000 32     AUTHOR:
0000 33
0000 34     C. FRANKS      05-OCT-1979
0000 35
0000 36     MODIFIED BY:
0000 37
0000 38     V03-008 WHM0001      Bill Matthews      15-May-1984
0000 39     Added MicroVAX I/QBUS support.
0000 40
0000 41     V03-007 RAS0300      Ron Schaefer      27-Apr-1984
0000 42     Add DEV$M_NNM characteristic to DECHAR2 so that these
0000 43     devices will have the 'node$' prefix.
0000 44
0000 45     V03-006 PRD0033      Paul R. DeStefano  09-Sep-1983
0000 46     Added EXE$LCLDSKVALID to function decision table.
0000 47
0000 48     V03-005 ROW0211      Ralph O. Weber    16-AUG-1983
0000 49     Change device-dependent UCB definition base from UCB$W_BCR+2
0000 50     to UCB$K_LCL_DISK_LENGTH.
0000 51
0000 52     V03-004 KDM0059      Kathleen D. Morse  14-Jul-1983
0000 53     Change time-wait loops to use new TIMEDWAIT macro.
0000 54
0000 55     V03-003 PRD0020      Paul R. DeStefano  26-Apr-1983
0000 56     Modified FATALERR routine to return $$$_PARITY only for
0000 57     errors that possibly indicate bad media. All other error
```



DLDRIVER  
V04-000

- VAX/VMS RL11/RL01,RL02 DISK DRIVER<sup>1 5</sup>

16-SEP-1 4 00:17:29 VAX/VMS Macro V04-00  
5-SEP- 4 00:12:24 [DRIVER.SRC]DLDRIVER.MAR;1

Page 2  
(1)

0000	58	:	conditions which formerly returned SSS_PARITY now return
0000	59	:	SSS_CNTLERR.
0000	60	:	
0000	61	:	V03-002 KDM0002 Kathleen D. Morse 28-Jun-1982
0000	62	:	Added \$DYNDEF.
0000	63	:	
0000	64	:	V03-001 KTA0100 Kerbey T. Altmann 07-Jun-1982
0000	65	:	Add code to set UCBSL_MEDIA_ID.
0000	66	:	
0000	67	:	**

```

0000 69 : ABSTRACT:
0000 70 :
0000 71 :     THIS MODULE CONTAINS THE TABLES AND ROUTINES NECESSARY TO
0000 72 :     PERFORM ALL DEVICE-DEPENDENT PROCESSING OF AN I/O REQUEST
0000 73 :     FOR RL11/RL01,RL02 DISK TYPES ON A VAX/VMS SYSTEM.
0000 74 :
0000 75 :     THE DISKS HAVE THE FOLLOWING PHYSICAL GEOMETRY:
0000 76 :
0000 77 :           # CYL      TRACKS/      SECTORS/      BYTES/      MAXIMUM
0000 78 :           # CYL      CYLINDER    TRACK        SECTOR      BLOCKS
0000 79 :
0000 80 :     RL01      256        2          40         256        10240
0000 81 :     RL02      512        2          40         256        20480
0000 82 :
0000 83 :     SINCE THE SECTOR SIZE IS ONLY 1/2 BLOCK, LOGICAL TO PHYSICAL
0000 84 :     CONVERSION OF THE DISK ADDRESS IS DONE IN THE DRIVER STARTIO
0000 85 :     ROUTINE RATHER THAN IN THE IOC$CVTLOGPHY FDT ROUTINE.
0000 86 :
0000 87 :     OVERLAPPED SEEKS ARE NOT ATTEMPTED BECAUSE THE DEVICE DOES
0000 88 :     NOT INTERRUPT AT THE COMPLETION OF A SEEK.
0000 89 :
0000 90 :     ALSO, THE DEVICE DOES NOT PERFORM AN IMPLICIT SEEK WHEN PERFORMING
0000 91 :     A READ OR WRITE FUNCTION,SO SEEK FUNCTIONS ARE ISSUED BY THIS
0000 92 :     DRIVER WHERE NECESSARY PRIOR TO ISSUING A READ OR WRITE FUNCTION.
0000 93 :     THE READ OR WRITE FUNCTION IS THEN ISSUED AS SOON AS THE RL11
0000 94 :     CONTROLLER COMES READY (WHILE THE SEEK IS IN PROGRESS), AND A
0000 95 :     WAIT FOR INTERRUPT (UPON COMPLETION OF THE READ OR WRITE) IS
0000 96 :     ISSUED. IF A SEEK FUNCTION IS REQUESTED SEPARATELY FROM A READ OR
0000 97 :     WRITE, A DUMMY READ HEADER FUNCTION IS ISSUED FOLLOWING THE SEEK
0000 98 :     FUNCTION AND A WAIT FOR INTERRUPT (UPON COMPLETION OF THE READ
0000 99 :     HEADER) IS ISSUED.
0000 100 :
0000 101 :     THE IOSX INHSEEK FUNCTION MODIFIER IS TREATED AS A NO-OP BY
0000 102 :     THIS DRIVER, SINCE AN EXPLICIT SEEK IS NECESSARY FOR THE RL02
0000 103 :     TO TRANSFER DATA PROPERLY.
0000 104 :
0000 105 :     THE RL'S DO NOT READ OR WRITE BEYOND THE END OF TRACK (THEY DO NOT
0000 106 :     AUTOMATICALLY SEEK THE NEXT TRACK), SO ALL READ AND WRITE FUNCTIONS
0000 107 :     ARE BROKEN UP BY THIS DRIVER INTO PARTIAL TRANSFERS TO THE END OF
0000 108 :     TRACK, FOLLOWED BY A SEEK TO THE NEXT TRACK, THEN ANOTHER READ OR
0000 109 :     WRITE FUNCTION UNTIL THE TOTAL DATA TRANSFER IS COMPLETE.
0000 110 :
0000 111 : --

```



```
0000 113      .SBTTL  EXTERNAL AND LOCAL DEFINITIONS
0000 114
0000 115      :
0000 116      : EXTERNAL SYMBOLS
0000 117      :
0000 118
0000 119      $ADPDEF      ;DEFINE ADAPTER CONTROL BLOCK
0000 120      $CRBDEF      ;DEFINE CHANNEL REQUEST BLOCK
0000 121      $DCDEF       ;DEFINE DEVICE CLASS
0000 122      $DDBDEF      ;DEFINE DEVICE DATA BLOCK
0000 123      $DEVDEF      ;DEFINE DEVICE CHARACTERISTICS
0000 124      $DPTDEF      ;DEFINE DRIVER PROLOGUE TABLE
0000 125      $DYNDEF      ;DEFINE DYNAMIC DATA STRUCTURE TYPES
0000 126      $EMBDEF      ;DEFINE ERROR MESSAGE BUFFER
0000 127      $IDBDEF      ;DEFINE INTERRUPT DATA BLOCK
0000 128      $IODEF       ;DEFINE I/O FUNCTION CODES
0000 129      $IRPDEF      ;DEFINE I/O REQUEST PACKET
0000 130      $PRDEF       ;DEFINE PROCESSOR REGISTERS
0000 131      $PTEDEF      ;DEFINE SYSTEM PTES
0000 132      $SSDEF       ;DEFINE SYSTEM STATUS CODES
0000 133      $UCBDEF      ;DEFINE UNIT CONTROL BLOCK
0000 134      $VADEF       ;DEFINE VIRTUAL ADDRESS BITS
0000 135      $VECDDEF     ;DEFINE INTERRUPT VECTOR BLOCK
0000 136
0000 137      :
0000 138      : LOCAL MACROS
0000 139      :
0000 140
0000 141      :
0000 142      : EXFUNCL
0000 143      : BRANCH TO SUBROUTINE WHICH REQUESTS CHANNEL (IF NOT ALREADY OWNED),
0000 144      : EXECUTES FCODE (OR R3) FUNCTION, AND BRANCHES TO BDST ON ERROR
0000 145      :
0000 146      .MACRO  EXFUNCL BDST,FCODE
0000 147      .IF NB  FCODE      ;IS FCODE NON-BLANK?
0000 148      MOVZBL  #CD'FCODE,R3 ;IF NB - SPECIFY FCODE FUNCTION
0000 149      .ENDC      ;IF B - SPECIFY FNTN IN EXISTING R3
0000 150      BSBW    FEXL      ;EXECUTE FUNCTION
0000 151      .BYTE   BDST-.-1  ;WHERE TO GO IF ERROR
0000 152      .ENDM
0000 153
0000 154      :
0000 155      : GENF
0000 156      : GENERATE FUNCTION TABLE ENTRY AND CASE TABLE INDEX SYMBOL
0000 157      :
0000 158      .MACRO  GENF FCODE
0000 159      CD'FCODE=-.FTAB/2
0000 160      .WORD   FCODE!RL_CS_M_IE ;FCODE WITH INT ENABLE BIT
0000 161      .ENDM
0000 162
0000 163      :
0000 164      : CKPWR
0000 165      : DISABLE INTERRUPTS, CHECK IF POWER HAS FAILED,
0000 166      : AND PUT DEVICE UNIT NUMBER IN R2<9:8>
0000 167      :
0000 168      :
0000 169      .MACRO  CKPWR ?L1
0000      CLRL    R2          ;CLEAR R2 FOR UNIT NUMBER
```



```
0000 170      INSV      UCBSW_UNIT(R5),- ;PUT UNIT # IN R2<9:8>
0000 171      #8,#2,R2
0000 172      DSBINT    ;DISABLE INTERRUPTS
0000 173      BBC      #UCBSV_POWER,- ;IF CLR - NO POWER FAILURE
0000 174      UCBSW_STS(R5),L1
0000 175      ENBINT    ;POWER FAILURE - ENABLE INTERRUPTS
0000 176      BRW      RETREG ;EXIT
0000 177 L1:      ;RETURN FOR NO POWER FAILURE
0000 178      .ENDM
0000 179
0000 180
0000 181      ;
0000 182      ; LOCAL SYMBOLS
0000 183      ;
0000 184
00000004 0000 185 RL_NUM_REGS =4 ;NUMBER OF DEVICE REGISTERS
00000005 0000 186 RL_SLM =5 ;STATE=SEEK LINEAR MODE (READY TO GO)
000000C9 0000 187 UCBSB_DL_DCCHK =UCBSW_OFFSET+1 ;REDEFINE FOR DATA CHECK USE
0000 188
0000 189      ;
0000 190      ; UCB OFFSETS WHICH FOLLOW THE STANDARD UCB FIELDS
0000 191      ;
0000 192      $DEFINI UCB ;START OF UCB DEFINITIONS
0000 193
000000CC 0000 194 .=UCBSK_LCL_DISK_LENGTH ;BEGIN DEFINITIONS AT END OF UCB
00CC 195 $DEF UCBSW_DL_PBCR .BLKW 1 ;PARTIAL BYTE COUNT
00CE 196 $DEF UCBSW_DL_CS .BLKW 1 ;CONTROL STATUS REGISTER
00D0 197 $DEF UCBSW_DL_BA .BLKW 1 ;BUS ADDRESS REGISTER
00D2 198 $DEF UCBSW_DL_DA .BLKW 1 ;DISK ADDRESS REGISTER
00D4 199 $DEF UCBSW_DL_MP .BLKW 1 ;MULTIPURPOSE REGISTER
00D6 200 $DEF UCBSW_DL_DPN .BLKW 1 ;DATA PATH NUMBER
00D8 201 $DEF UCBSL_DL_SVAPTE ;SAVED SVAPTE OF THE USER'S BUFFER
00D8 202 $DEF UCBSL_DL_DPR .BLKL 1 ;DATAPATH REGISTER
00DC 203 $DEF UCBSL_DL_BUFADR ;USER BUFFER ADDRESS
00DC 204 $DEF UCBSL_DL_FMPR .BLKL 1 ;FINAL MAP REGISTER
00E0 205 $DEF UCBSA_DL_MOVRTN ;BUFFER MOVE ROUTINE ADDRESS
00E0 206 $DEF UCBSL_DL_PMPR .BLKL 1 ;PREVIOUS MAP REGISTER
00E4 207 $DEF UCBSB_DL_DPPE .BLKB 1 ;DATAPATH PURGE ERROR
00E5 208 $DEF UCBSW_DL_DB .BLKW 3 ;DATA BUFFER REGISTER
00EB 209 $DEF UCBSB_DL_XBA .BLKB 1 ;BUS ADDRESS EXTENSION BITS
00EC 210 $DEF UCBSW_DL_SBA .BLKW 1 ;SAVED BUFFER ADDRESS
00EE 211 $DEF UCBSA_DL_BUF_VA .BLKL 1 ;PHYSICAL BUFFER VIRTUAL ADDRESS
00F2 212 $DEF UCBSA_DL_BUF_PA .BLKL 1 ;PHYSICAL BUFFER PHYSICAL ADDRESS
00F6 213 $DEF UCBSW_DL_FLAGS .BLKW 1 ;FLAGS
00F8 214 $VIELD UCB,0,<- ;START THE FLAG DEFINITIONS
00F8 215 <DL_22BIT,,M>,- ;22 BIT ADDRESSING
00F8 216 <DL_MAPPING,,M>,- ;ADAPTER MAPPING
00F8 217 > ;END OF FLAG DEFINITIONS
00F8 218 $DEF UCBSK_DL_LEN .BLKW 1 ;LENGTH OF UCB
00FA 219 $EQU UCBSK_DL_BUFSZ 20 ;BUFFER SIZE = 40 SECTORS *
00FA 220 ;256 BYTES/SECTOR / 512 BYTES/PAGE
00FA 221 $DEFEND UCB ;END OF UCB DEFINITIONS
0000 222
0000 223      ;
0000 224      ; RL11/RL01 REGISTER OFFSETS FROM CSR ADDRESS
0000 225      ;
0000 226      $DEFINI RL ; START OF REGISTER DEFINITIONS
```



```
0000 227
0000 228 $DEF RL_CS .BLKW 1 ;CONTROL STATUS REGISTER (CSR)
0002 229 _VIELD RL_CS,0,<- ;START OF CSR BIT DEFINITIONS
0002 230 <DRDY,,M>,- ;DRIVE READY
0002 231 <FCODE,,3>,- ;FUNCTION CODE
0002 232 <XBA,,2>,- ;BUS ADDRESS EXTENSION BITS
0002 233 <IE,,M>,- ;INTERRUPT ENABLE
0002 234 <CRDY,,M>,- ;CONTROLLER READY
0002 235 <DS,,2>,- ;DRIVE SELECT
0002 236 <OPI,,M>,- ;OPERATION INCOMPLETE
0002 237 <CRC,,M>,- ;DATA CRC OR HEADER CRC
0002 238 <DLT,,M>,- ;DATA LATE OR HEADER NOT FOUND
0002 239 <NXM,,M>,- ;NON-EXISTENT MEMORY
0002 240 <DE,,M>,- ;DRIVE ERROR
0002 241 <CE,,M>,- ;COMPOSITE ERROR
0002 242 > ;END CSR BIT DEFINITIONS
0002 243
0002 244 $DEF RL_BA .BLKW 1 ;BUS ADDRESS REGISTER (BAR)
0004 245
0004 246 $DEF RL_DA .BLKW 1 ;DISK ADDRESS REGISTER (DAR)
0006 247 _VIELD RL_DA,0,<- ;START OF DAR BIT DEFINITIONS
0006 248 <MRK,,M>,- ;MARK (ALWAYS 1)
0006 249 <STS,,M>,- ;GET STATUS
0006 250 <,1>,- ;RESERVED BIT
0006 251 <RST,,M>,- ;RESET
0006 252 <,12>,- ;RESERVED BITS
0006 253 > ;END OF DAR BIT DEFINITIONS
0006 254
0006 255 $DEF RL_MP .BLKW 1 ;MULTIPURPOSE REGISTER (MPR)
0008 256 _VIELD RL_MP,0,<- ;START OF MPR BIT DEFINITIONS
0008 257 <STA,,3>,- ;DRIVE STATE
0008 258 <BH,,M>,- ;BRUSH HOME
0008 259 <HO,,M>,- ;HEADS OUT
0008 260 <CO,,M>,- ;COVER OPEN
0008 261 <HS,,M>,- ;HEAD SELECT
0008 262 <TYP,,M>,- ;DRIVE TYPE
0008 263 <DSE,,M>,- ;DRIVE SELECT ERROR
0008 264 <VC,,M>,- ;VOLUME CHECK
0008 265 <WGE,,M>,- ;WRITE GATE ERROR
0008 266 <SPE,,M>,- ;SPIN ERROR
0008 267 <SKTO,,M>,- ;SEEK TIME OUT
0008 268 <WL,,M>,- ;WRITE LOCK
0008 269 <CHE,,M>,- ;CURRENT HEAD ERROR
0008 270 <WDE,,M>,- ;WRITE DATA ERROR
0008 271 > ;END MPR BIT DEFINITIONS
0008 272
0008 273 $DEF RL_BAE .BLKW 1 ;BUS ADDRESS EXTENSION REGISTER(BAE)
000A 274
000A 275 $DEFEND RL ;END RL11/RL01 REGISTER DEFINITIONS
0000 276
0000 277 ;
0000 278 ; HARDWARE FUNCTION CODES
0000 279 ;
00000000 0000 280 F_NOP=0*2 ;NO OPERATION
00000000 0000 281 F_UNLOAD=F_NOP ;NO OPERATION
00000006 0000 282 F_SEEK=3*2 ;SEEK CYLINDER
00000000 0000 283 F_RECAL=F_NOP ;NO OPERATION
```

DLDRIVER  
V04-000

- VAX/VMS RL11/RL01,RL02 DISK DRIVER<sup>N 5</sup>  
EXTERNAL AND LOCAL DEFINITIONS

16-SEP-1984 00:17:29 VAX/VMS Macro V04-00  
5-SEP-1984 00:12:24 [DRIVER.SRC]DLDRIVER.MAR;1

Page 7  
(1)

00000004	0000	284	F_DRVCLR=2*2
00000000	0000	285	F_RELEASE=F_NOP
00000000	0000	286	F_OFFSET=F_NOP
00000000	0000	287	F_RETCENTER=F_NOP
00000004	0000	288	F_PACKACK=2*2
00000000	0000	289	F_SEARCH=F_NOP
00000002	0000	290	F_WRITECHECK=1*2
0000000A	0000	291	F_WRITEDATA=5*2
00000000	0000	292	F_WRITEHEAD=F_NOP
0000000C	0000	293	F_READDATA=6*2
00000008	0000	294	F_READHEAD=4*2
00000000	0000	295	F_AVAILABLE=F_NOP
00000004	0000	296	F_GETSTATUS=2*2
	0000	297	

;DRIVE CLEAR (GET STATUS)  
;NO OPERATION  
;NO OPERATION  
;NO OPERATION  
;PACK ACKNOWLEDGE (SET VOLUME VALID)  
;NO OPERATION  
;WRITE CHECK  
;WRITE DATA  
;NO OPERATION  
;READ DATA  
;READ HEADER  
;NO OPERATION  
;GET STATUS (DRIVER INTERNAL USE)



```
0000 299      .SBTTL  STANDARD TABLES
0000 300
0000 301      :
0000 302      : DRIVER PROLOGUE TABLE
0000 303      :
0000 304      : THE DPT DESCRIBES DRIVER PARAMETERS AND I/O DATABASE FIELDS
0000 305      : THAT ARE TO BE INITIALIZED DURING DRIVER LOADING AND RELOADING
0000 306      :
0000 307      :
0000 308      DPTAB      -      ;DPT CREATION MACRO
0000 309      END=DL END,-      ;END OF DRIVER LABEL
0000 310      ADAPTER=UBA,-      ;ADAPTER TYPE = UNIBUS
0000 311      FLAGS=DPT$M_SVP,-      ;SYSTEM PAGE TABLE ENTRY REQUIRED
0000 312      UCBSIZE=UCB$K_DL_LEN,-      ;LENGTH OF UCB
0000 313      NAME=DLDRIVER      ;DRIVER NAME
0038 314
0038 315      DPT_STORE INIT      ;START CONTROL BLOCK INIT VALUES
0038 316      DPT_STORE DDB,DDB$$_ACPD,L,<^A\F11\>      ;DEFAULT ACP NAME
003F 317      DPT_STORE DDB,DDB$$_ACPD+3,B,DDB$$_K_CART      ;ACP CLASS
0043 318      DPT_STORE UCB,UCB$$_FIPL,B,8      ;FORK IPL
0047 319      DPT_STORE UCB,UCB$$_DEVCHAR,L,-      ;DEVICE CHARACTERISTICS
0047 320      <DEV$M_FOD-      ;FILES ORIENTED
0047 321      DEV$M_DIR-      ;DIRECTORY STRUCTURED
0047 322      DEV$M_AVL-      ;AVAILABLE
0047 323      DEV$M_ELG-      ;ERROR LOGGING
0047 324      DEV$M_SHR-      ;SHAREABLE
0047 325      DEV$M_IDV-      ;INPUT DEVICE
0047 326      DEV$M_ODV-      ;OUTPUT DEVICE
0047 327      DEV$M_RND>      ;RANDOM ACCESS
004E 328      DPT_STORE UCB,UCB$$_DEVCHAR2,L,-      ;DEVICE CHARACTERISTICS
004E 329      <DEV$M_NNM>      ;PREFIX NAME WITH 'node$'
0055 330      DPT_STORE UCB,UCB$$_DEVCLASS,B,DDB$$_K_DISK      ;DEVICE CLASS
0059 331      DPT_STORE UCB,UCB$$_DEVBUFSIZ,W,512      ;DEFAULT BUFFER SIZE
005E 332      DPT_STORE UCB,UCB$$_SECTORS,B,40      ;NUMBER OF SECTORS PER TRACK
0062 333      DPT_STORE UCB,UCB$$_TRACKS,B,2      ;NUMBER OF TRACKS PER CYLINDER
0066 334      DPT_STORE UCB,UCB$$_DIPL,B,21      ;DEVICE IPL
006A 335      DPT_STORE UCB,UCB$$_ERTMAX,B,8      ;MAX ERROR RETRY COUNT
006E 336      DPT_STORE UCB,UCB$$_DEVSTS,W,-      ;INHIBIT LOG TO PHYS CONVERSION IN FDT
006E 337      <UCB$M_NOCNVRT>      ;...
0073 338
0073 339      DPT_STORE REINIT      ;START CONTROL BLOCK RE-INIT VALUES
0073 340      DPT_STORE CRB,CRB$$_INTD+4,D,DL_INT      ;INTERRUPT SERVICE ROUTINE ADDRESS
0078 341      DPT_STORE CRB,CRB$$_INTD+VEC$$_INITIAL,-      ;CONTROLLER INIT ADDRESS
0078 342      D,DL_RL11_INIT      ;...
007D 343      DPT_STORE CRB,CRB$$_INTD+VEC$$_UNITINIT,-      ;UNIT INIT ADDRESS
007D 344      D,DL_RLOX_INIT      ;...
0082 345      DPT_STORE DDB,DDB$$_DDT,D,DL$DDT      ;DDT ADDRESS
0087 346
0087 347      DPT_STORE END      ;END OF INITIALIZATION TABLE
0000 348
0000 349      :
0000 350      : DRIVER DISPATCH TABLE
0000 351      :
0000 352      : THE DDT LISTS ENTRY POINTS FOR DRIVER SUBROUTINES WHICH ARE
0000 353      : CALLED BY THE OPERATING SYSTEM.
0000 354      :
0000 355
```



```
0000 356      DDTAB - ;DDT CREATION MACRO
0000 357      DEVNAM=DL,- ;NAME OF DEVICE
0000 358      START=DL $STARTIO,- ;START I/O ROUTINE
0000 359      UNSOLIC=DL UNSOLNT,- ;UNSOLICITED INTERRUPT
0000 360      FUNCTB=DL FUNCTABLE,- ;FUNCTION DECISION TABLE
0000 361      CANCEL=0,- ;CANCEL=NO-OP FOR FILES DEVICE
0000 362      REGDMP=DL REGDUMP,- ;REGISTER DUMP ROUTINE
0000 363      DIAGBF=<<<RL_NUM_REGS+5+5+3+1>*4>,- ;BYTES IN DIAG BUFFER
0000 364      ERLGBF=<<<<RL_NUM_REGS+5+1>*4>+EMBSL_DV_REGSAV> ;BYTES IN
0038 365      ;ERROR LOG BUFFER
0038 366      :
0038 367      : DIAGNOSTIC BUFFER SIZE = <<<4 RL02 REGISTER LONGWORDS + 5 UCB FIELD LONGWORDS
0038 368      : + 5 IOC$DIAGBUFILL LONGWORDS + 3 BUFFER ALLOCATION
0038 369      : LONGWORDS + 1 LONGWORD FOR # REGISTERS IN DL_REGDUMP>
0038 370      : * 4 BYTES/LONGWORD>
0038 371      :
0038 372      : ERROR LOG BUFFER SIZE = <<<<4 RL02 REGISTER LONGWORDS + 5 UCB FIELD LONGWORDS
0038 373      : + 1 LONGWORD FOR # REGISTERS IN DL_REGDUMP>
0038 374      : * 4 BYTES/LONGWORD> + BYTES NEEDED FOR ERROR LOGGER
0038 375      : TO SAVE SOFTWARE REGISTERS>
0038 376      :
0038 377      :
0038 378      :
0038 379      : HARDWARE FUNCTION CODE TABLE
0038 380      :
0038 381      : THIS TABLE MERGES THE FUNCTION CODE BITS WITH THE
0038 382      : INTERRUPT ENABLE BIT AND GENERATES THE CASE TABLE
0038 383      : INDEX SYMBOL.
0038 384      :
0038 385      FTAB: GENF F_NOP ;NO-OP
003A 386      GENF F_UNLOAD ;UNLOAD VOLUME (NOP)
003C 387      GENF F_SEEK ;SEEK
003E 388      GENF F_RECAL ;RECALIBRATE (NOP)
0040 389      GENF F_DRVCLR ;DRIVE CLEAR (RESET & GET STATUS)
0042 390      GENF F_RELEASE ;RELEASE PORT (NOP)
0044 391      GENF F_OFFSET ;OFFSET HEADS (NOP)
0046 392      GENF F_RETCENTER ;RETURN HEADS TO CENTERLINE (NOP)
0048 393      GENF F_PACKACK ;PACK ACKNOWLEDGE (RESET & GET STATUS)
004A 394      GENF F_SEARCH ;SEARCH (NOP)
004C 395      GENF F_WRITECHECK ;WRITE CHECK
004E 396      GENF F_WRITEDATA ;WRITE DATA
0050 397      GENF F_READDATA ;READ DATA
0052 398      GENF F_WRITEHEAD ;WRITE HEADERS (NOP)
0054 399      GENF F_READHEAD ;READ HEADERS
0056 400      GENF F_NOP ;place holder
0058 401      GENF F_NOP ;place holder
005A 402      GENF F_AVAILABLE ;AVAILABLE
005C 403
```



```
005C 405 :  
005C 406 : FUNCTION DECISION TABLE  
005C 407 :  
005C 408 : THE FDT LISTS VALID FUNCTION CODES, SPECIFIES WHICH  
005C 409 : CODES ARE BUFFERED, AND DESIGNATES SUBROUTINES TO  
005C 410 : PERFORM PREPROCESSING FOR PARTICULAR FUNCTIONS.  
005C 411 :  
005C 412 :  
005C 413 DL_FUNCTABLE:  
005C 414 FUNCTAB  
005C 415 <NOP,-  
005C 416 UNLOAD,-  
005C 417 SEEK,-  
005C 418 DRVCLR,-  
005C 419 PACKACK,-  
005C 420 SENSECHAR,-  
005C 421 SETCHAR,-  
005C 422 SENSEMODE,-  
005C 423 SETMODE,-  
005C 424 WRITECHECK,-  
005C 425 READHEAD,-  
005C 426 READLBLK,-  
005C 427 WRITELBLK,-  
005C 428 READPBLK,-  
005C 429 WRITEPBLK,-  
005C 430 READVBLK,-  
005C 431 WRITEVBLK,-  
005C 432 AVAILABLE,-  
005C 433 ACCESS,-  
005C 434 ACPCONTROL,-  
005C 435 CREATE,-  
005C 436 DEACCESS,-  
005C 437 DELETE,-  
005C 438 MODIFY,-  
005C 439 MOUNT-  
005C 440 >  
0064 441 FUNCTAB  
0064 442 <NOP,-  
0064 443 UNLOAD,-  
0064 444 SEEK,-  
0064 445 DRVCLR,-  
0064 446 PACKACK,-  
0064 447 SENSECHAR,-  
0064 448 SETCHAR,-  
0064 449 SENSEMODE,-  
0064 450 SETMODE,-  
0064 451 AVAILABLE,-  
0064 452 ACCESS,-  
0064 453 ACPCONTROL,-  
0064 454 CREATE,-  
0064 455 DEACCESS,-  
0064 456 DELETE,-  
0064 457 MODIFY,-  
0064 458 MOUNT-  
0064 459 >  
006C 460 FUNCTAB DL_ALIGN,-  
006C 461 <READHEAD,-
```

```
:LIST LEGAL FUNCTIONS  
: NO-OP  
: UNLOAD  
: SEEK  
: DRIVE CLEAR  
: PACK ACKNOWLEDGE  
: SENSE CHARACTERISTICS  
: SET CHARACTERISTICS  
: SENSE MODE  
: SET MODE  
: WRITE CHECK  
: READ HEADER  
: READ LOGICAL BLOCK  
: WRITE LOGICAL BLOCK  
: READ PHYSICAL BLOCK  
: WRITE PHYSICAL BLOCK  
: READ VIRTUAL BLOCK  
: WRITE VIRTUAL BLOCK  
: AVAILABLE  
: ACCESS FILE / FIND DIRECTORY ENTRY  
: ACP CONTROL FUNCTION  
: CREATE FILE AND/OR DIRECTORY ENTRY  
: DEACCESS FILE  
: DELETE FILE AND/OR DIRECTORY ENTRY  
: MODIFY FILE ATTRIBUTES  
: MOUNT VOLUME  
  
:BUFFERED FUNCTIONS  
: NO-OP  
: UNLOAD  
: SEEK  
: DRIVE CLEAR  
: PACK ACKNOWLEDGE  
: SENSE CHARACTERISTICS  
: SET CHARACTERISTICS  
: SENSE MODE  
: SET MODE  
: AVAILABLE  
: ACCESS FILE / FIND DIRECTORY ENTRY  
: ACP CONTROL FUNCTION  
: CREATE FILE AND/OR DIRECTORY ENTRY  
: DEACCESS FILE  
: DELETE FILE AND/OR DIRECTORY ENTRY  
: MODIFY FILE ATTRIBUTES  
: MOUNT VOLUME  
  
:TEST ALIGNMENT FUNCTIONS  
: READ HEADER
```



006C	462	READBLK,-	: READ LOGICAL BLOCK
006C	463	READPBLK,-	: READ PHYSICAL BLOCK
006C	464	READVBLK,-	: READ VIRTUAL BLOCK
006C	465	WRITECHECK,-	: WRITE CHECK
006C	466	WRITEBLK,-	: WRITE LOGICAL BLOCK
006C	467	WRITEPBLK,-	: WRITE PHYSICAL BLOCK
006C	468	WRITEVBLK-	: WRITE VIRTUAL BLOCK
006C	469	>	
0078	470	FUNCTAB +ACPSREADBLK,-	: READ FUNCTIONS
0078	471	<READHEAD,-	: READ HEADER
0078	472	READBLK,-	: READ LOGICAL BLOCK
0078	473	READPBLK,-	: READ PHYSICAL BLOCK
0078	474	READVBLK-	: READ VIRTUAL BLOCK
0078	475	>	
0084	476	FUNCTAB +ACPSWRITEBLK,-	: WRITE FUNCTIONS
0084	477	<WRITECHECK,-	: WRITE CHECK
0084	478	WRITEBLK,-	: WRITE LOGICAL BLOCK
0084	479	WRITEPBLK,-	: WRITE PHYSICAL BLOCK
0084	480	WRITEVBLK-	: WRITE VIRTUAL BLOCK
0084	481	>	
0090	482	FUNCTAB +ACPSACCESS,-	: ACCESS FUNCTIONS
0090	483	<ACCESS,-	: ACCESS FILE / FIND DIRECTORY ENTRY
0090	484	CREATE-	: CREATE FILE AND/OR DIRECTORY ENTRY
0090	485	>	
009C	486	FUNCTAB +ACPSDEACCESS,-	: DEACCESS FUNCTION
009C	487	<DEACCESS-	: DEACCESS FILE
009C	488	>	
00A8	489	FUNCTAB +ACPSMODIFY,-	: MODIFY FUNCTIONS
00A8	490	<ACPCONTROL,-	: ACP CONTROL FUNCTION
00A8	491	DELETE,-	: DELETE FILE AND/OR DIRECTORY ENTRY
00A8	492	MODIFY-	: MODIFY FILE ATTRIBUTES
00A8	493	>	
00B4	494	FUNCTAB +ACPSMOUNT,-	: MOUNT FUNCTION
00B4	495	<MOUNT-	: MOUNT VOLUME
00B4	496	>	
00C0	497	FUNCTAB +EXESLCLDSKVALID,-	: LOCAL DISK VALID FUNCTIONS
00C0	498	<UNLOAD,-	: UNLOAD VOLUME
00C0	499	AVAILABLE,-	: UNIT AVAILABLE
00C0	500	PACKACK-	: PACK ACKNOWLEDGE
00C0	501	>	
00CC	502	FUNCTAB +EXESZEROPARM,-	: ZERO PARAMETER FUNCTIONS
00CC	503	<NOP,-	: NO-OP
00CC	504	UNLOAD,-	: UNLOAD
00CC	505	DRVCLR,-	: DRIVE CLEAR
00CC	506	PACKACK,-	: PACK ACKNOWLEDGE
00CC	507	AVAILABLE,-	: AVAILABLE
00CC	508	>	
00D8	509	FUNCTAB +EXESONEPARM,-	: ONE PARAMETER FUNCTION
00D8	510	<SEEK-	: SEEK
00D8	511	>	
00E4	512	FUNCTAB +EXESSENSEMODE,-	: SENSE FUNCTIONS
00E4	513	<SENSECHAR,-	: SENSE CHARACTERISTICS
00E4	514	SENSEMODE-	: SENSE MODE
00E4	515	>	
00F0	516	FUNCTAB +EXESSETCHAR,-	: SET FUNCTIONS
00F0	517	<SETCHAR,-	: SET CHARACTERISTICS
00F0	518	SETMODE-	: SET MODE



DLDRIVER  
V04-000

- VAX/VMS RL11/RL01,RL02 DISK DRIVER<sup>F 6</sup>  
STANDARD TABLES

00F0 519

>

16-SEP-1984 00:17:29  
5-SEP-1984 00:12:24

VAX/VMS Macro V04-00  
[DRIVER.SRC]DLDRIVER.MAR;1

Page 12  
(1)

```
00FC 521
00FC 522 .SBTTL CONTROLLER INITIALIZATION ROUTINE
00FC 523 :++
00FC 524
00FC 525 FUNCTIONAL DESCRIPTION:
00FC 526
00FC 527 THIS ROUTINE IS A NO-OP FOR THE RL11 BUT MUST BE INCLUDED
00FC 528 SINCE IT IS CALLED WHEN THE RL02 IS BOOTED AS A SYSTEM DEVICE.
00FC 529
00FC 530 THE OPERATING SYSTEM CALLS THIS ROUTINE:
00FC 531 - AT SYSTEM STARTUP
00FC 532 - DURING DRIVER LOADING
00FC 533 - DURING RECOVERY FROM POWER FAILURE
00FC 534
00FC 535 INPUTS:
00FC 536
00FC 537 R4 - CSR ADDRESS (DEVICE CONTROL STATUS REGISTER)
00FC 538 R5 - IDB ADDRESS (INTERRUPT DATA BLOCK)
00FC 539 R6 - DDB ADDRESS (DEVICE DATA BLOCK)
00FC 540 R8 - CRB ADDRESS (CHANNEL REQUEST BLOCK)
00FC 541 ALL INTERRUPTS ARE LOCKED OUT
00FC 542
00FC 543 OUTPUTS:
00FC 544
00FC 545 ALL REGISTERS EXCEPT R0-R3 ARE PRESERVED.
00FC 546 CONTROL IS RETURNED TO THE CALLER.
00FC 547
00FC 548 :--
00FC 549
00FC 550 DL_RL11_INIT: ;CONTROLLER INITIALIZATION
00FC 551
00FC 552 ; FOR MICROVAX I, ALLOCATE A PHYSICALLY CONTIGUOUS BUFFER
00FC 553 ; AREA FOR PERFORMING I/O.
00FC 554
00FC 555 CPUDISP <<790,20$>,-
00FC 556 <785,20$>,-
00FC 557 <780,20$>,-
00FC 558 <750,20$>,-
00FC 559 <730,20$>,-
00FC 560 <UV1,10$>> ;FOR MICROVAX I, ALLOCATE BUFFER AREA
012E 561 ;FOR ALL OTHERS, SKIP BUFFER AREA
012E 562
51 14 3C 012E 563 10$: MOVZWL #UCB$K_DL_BUFSZ,R1 ;LOAD SIZE OF BUFFER
00000000'GF 16 0131 564 JSB G^EXE$XLOPHYCNTG ;ALLOCATE PHYSICALLY-CONTIGUOUS MEMORY
05 50 E9 0137 565 BLBC R0,20$ ;EXIT ON ERROR
10 A8 52 D0 013A 566 MOVL R2,CRB$L_AUXSTRUC(R8) ;GET BUFFER VIRTUAL ADDRESS
05 013E 567 RSB ;RETURN TO CALLER
013F 568
10 A8 D4 013F 569 20$: CLRL CRB$L_AUXSTRUC(R8) ;INDICATE MEMORY ALLOCATION FAILURE
05 0142 570 RSB ;RETURN TO CALLER
```



```
0143 572 .SBTTL UNIT INITIALIZATION ROUTINE
0143 573
0143 574 :++
0143 575
0143 576 DL_RLOX_INIT - UNIT INITIALIZATION ROUTINE
0143 577
0143 578 FUNCTIONAL DESCRIPTION:
0143 579
0143 580 THIS ROUTINE READIES THE RL01/RL02 UNITS FOR I/O OPERATIONS.
0143 581
0143 582 THE OPERATING SYSTEM CALLS THIS ROUTINE:
0143 583 - AT SYSTEM STARTUP
0143 584 - DURING DRIVER LOADING
0143 585 - DURING RECOVERY FROM POWER FAILURE
0143 586
0143 587 INPUTS:
0143 588
0143 589 R4 - CSR ADDRESS (CONTROLLER STATUS REGISTER)
0143 590 R5 - UCB ADDRESS (UNIT CONTROL BLOCK)
0143 591
0143 592 OUTPUTS:
0143 593
0143 594 THE DRIVE UNIT IS RESET, UCB FIELDS ARE INITIALIZED, AND THE
0143 595 ROUTINE WAITS FOR ONLINE UNITS TO SPIN UP. ALL REGISTERS
0143 596 EXCEPT R0-R3 ARE PRESERVED.
0143 597
0143 598 :--
0143 599
0143 600 DL_RLOX_INIT: ;RL01/RL02 UNIT INITIALIZATION
0143 601 MOVW #1@UCB$V_DL_MAPPING,- ; DEFAULT TO ADAPTER MAPPING
0145 602 UCB$W_DL_FLAGS(R5) ; AND 18 BIT ADDRESSING
0148 603
0148 604 ; SET CPU DEPENDENT UCB FLAGS FOR DL
0148 605
0148 606 CPUDISP <<790,10$>,-
0148 607 <785,10$>,-
0148 608 <780,10$>,-
0148 609 <750,10$>,-
0148 610 <730,10$>,-
0148 611 <UV1,5$>>
017A 612 5$: MOVW #1@UCB$V_DL_22BIT,- ; FOR MICROVAX I 22 BIT
017C 613 UCB$W_DL_FLAGS(R5) ; ADDRESSING AND NO ADAPTER MAPPING
017F 614 10$: MOVZWL UCB$W_STS(R5),R3 ; SAVE CURRENT UNIT STATUS
0183 615 BICW #UCB$M_ONLINE!UCB$M_VALID,- ; ASSUME OFFLINE/INVALID
0187 616 UCB$W_STS(R5) ;...
0189 617
0189 618 ; WAIT FOR CONTROLLER (6 SECONDS MAX) IF CHANNEL IS BUSY WITH ANOTHER UNIT
0189 619
0189 620
0189 621
0189 622 MOVL UCB$L_CRB(R5),R0 ; GET CRB ADDRESS
018D 623 BBC #CRB$V_BSY,CRB$B_MASK(R0),20$ ; IF CLEAR - CHANNEL NOT BUSY
0192 624 TIMEDWAIT TIME=#600*1000,- ; 6 SECOND WAIT LOOP
0192 625 INS1=<TSTB RL CS(R4)>,- ; IS CONTROLLER READY
0192 626 INS2=<BLSS 15$>,- ; IF LSS - YES
0192 627 DONELBL=15$ ; LABEL TO EXIT WAIT LOOP
0188 628 BLBC R0,25$ ; TIME EXPIRED - EXIT
```



```
01BB 629 :  
01BB 630 : GET CURRENT DRIVE STATUS AND RESET DRIVE  
01BB 631 :  
01BB 632 :  
51 08 08 54 51 D4 01BB 633 20$: MOVW #RL_DA_M_RST!- :PUT RESET AND GET STATUS IN DAR  
64 04 04 51 A5 F0 01BF 634 :RL_DA_M_STS!RL_DA_M_MRK,RL_DA(R4) :...  
0095 30 01C1 635 :CLEAR R1 FOR UNIT NUMBER  
64 95 01C7 636 :UCBSW_UNIT(R5),#8,#8,R1 :GET UNIT NUMBER  
24 18 01CB 637 :R1,#F-GETSTATUS,RL_CS(R4) :EXECUTE GET STATUS FUNCTION  
01CE 638 :DL_WAIT :WAIT FOR CONTROLLER  
01D0 639 :RL_CS(R4) :WAS CONTROLLER READY?  
01D2 640 :BGEQ 25$ :IF GEQ - NO  
01D2 641 :  
01D2 642 :  
01D2 643 : CLASSIFY DRIVE TYPE  
01D2 644 :  
01D2 645 :  
2324C001 8F D0 01D2 646 :MOVL #X2324C001,-  
008C C5 01D8 647 :UCBSL_MEDIA_ID(R5) :SET MEDIA IDENT 'DL RL01'  
06 A4 0080 8F B3 01DB 648 :BITW #RL_MP_M_TYP,RL_MP(R4) :IS DRIVE TYPE = RL02?  
15 12 01E1 649 :BNEQ 30$ :IF NEQ - YES  
09 90 01E3 650 :MOVB S^#DTS_RL01,-  
41 A5 01E5 651 :UCBSB_DEVTYPE(R5) :SET RL01 DEVICE TYPE  
46 A5 0100 8F B0 01E7 652 :MOVW #256,UCBSW_CYLINDERS(R5) :SET NUMBER OF RL01 CYLINDERS  
00B0 C5 2800 8F 3C 01ED 653 :MOVZWL #10240,UCBSL_MAXBLOCK(R5) :SET MAX RL01 BLOCK NUMBER  
17 11 01F4 654 :BRB 40$  
6A 11 01F6 655 :  
01F8 656 25$: BRB 70$ :BRANCH TO COMMON EXIT  
0A 90 01F8 657 :  
41 A5 01FA 658 30$: MOVB S^#DTS_RL02,-  
46 A5 0200 8F B0 01FC 659 :UCBSB_DEVTYPE(R5) :SET RL02 DEVICE TYPE  
00B0 C5 5000 8F 3C 0202 660 :MOVW #512,UCBSW_CYLINDERS(R5) :SET NUMBER OF RL02 CYLINDERS  
008C C5 D6 0209 661 :MOVZWL #20480,UCBSL_MAXBLOCK(R5) :SET MAX RL02 BLOCK NUMBER  
16 53 0B E1 020D 662 :INCL UCBSL_MEDIA_ID(R5) :SET MEDIA IDENT 'DL RL02'  
0211 663 40$: BBC #UCBSV_VALID,R3,60$ :Branch around wait for drive to spinup  
0211 664 : if the drive did NOT have a VALID  
0211 665 : volume on it before POWER failure.  
0211 666 :  
0211 667 :  
0211 668 : INITIALIZE UCB FIELDS AND WAIT FOR ONLINE UNITS TO SPIN UP  
0211 669 :  
0211 670 :  
64 01 B3 0211 671 45$: BITW #RL_CS_M_DRDY,RL_CS(R4) : Is drive ready?  
0B 12 0214 672 :BNEQ 50$ :IF NEQ - YES  
00000000 GF 16 0216 673 :JSB G^EXESPWRTIMCHK :IS MAX TIME EXCEEDED?  
F2 50 E8 021C 674 :BLBS R0,45$ :IF LBS - NO, STILL MORE TIME NEEDED  
06 11 021F 675 :BRB 60$ :POWER UP TIME EXCEEDED  
0221 676 50$:  
64 A5 0800 8F A8 0221 677 :BISW #UCBSM_VALID,UCBSW_STS(R5) :SET UCB STATUS VOLUME VALID  
0227 678 :  
01 E0 0227 679 60$: BBS #UCBSV_DL_MAPPING,- :ADAPTER MAPPING?  
31 00F6 C5 0229 680 :UCBSW_DL_FLAGS(R5),65$ :IF BS YES  
51 24 A5 D0 022D 681 :MOVL UCBSL_CRB(R5),R1 :GET CRB ADDRESS  
52 10 A1 D0 0231 682 :MOVL CRBSL_AUXSTRU(R1),R2 :MEMORY ALLOC FAILURE DURING CTL INIT?  
2B 13 0235 683 :BEQL 70$ :IF EQL YES, LEAVE OFFLINE  
00EE C5 52 D0 0237 684 :MOVL R2,UCBSA_DL_BUF_VA(R5) :SAVE BUFFER'S VIRTUAL ADDRESS  
51 52 15 09 EF 023C 685 :EXTZV #VASV_VPN,#VASS_VPN,R2,R1 :GET VIRTUAL PAGE NUMBER OF BUFFER
```



DLDRIVER  
V04-000

- VAX/VMS RL11/RL01,RL02 DISK DRIVER<sup>J 6</sup>  
UNIT INITIALIZATION ROUTINE

16-SEP-1984 00:17:29 VAX/VMS Macro V04-00  
5-SEP-1984 00:12:24 [DRIVER.SRC]DLDRIVER.MAR;1

Page 16  
(1)

50	00000000*GF	D0	0241	686	MOVL	G*MMG\$GL_SPTBASE,R0	;GET BASE ADDRESS OF SPTS
	50 6041	D0	0248	687	MOVL	(R0)[R1],R0	;GET THE PTE CONTENTS
51	52	FFFFFE00 8F	CB	024C	688	BICL3	#^C<VASM BYTE>,R2,R1 ;GET BUFFER OFFSET (BA00-BA08)
				0254	689	ASSUME	PTES\$ PFN GE 13
51	0D 09 50	F0	0254	690	INSV	R0,#9,#13,R1	;COPY BA09-BA21
	00F2 C5 51	D0	0259	691	MOVL	R1,UCBSA_DL BUF_PA(R5)	;SAVE PHYSICAL ADDRESS OF BUFFER
	64 A5 10	AB	025E	692 65\$:	BISW	#UCBSM_ONLINE,UCBSW_STS(R5)	;SET UCB STATUS VOLUME VALID
		05	0262	693 70\$:	RSB		

```
0263 695 .SBTTL DRIVER SPECIFIC SUBROUTINES
0263 696 :
0263 697 : DL_WAIT - WAIT FOR CONTROLLER READY
0263 698 :
0263 699 : INPUTS:
0263 700 : R4 - DEVICE CSR ADDRESS
0263 701 :
0263 702 : FUNCTIONAL DESCRIPTION:
0263 703 :
0263 704 : THIS ROUTINE IS CALLED FROM THE DRIVER UNIT INITIALIZATION ROUTINE
0263 705 : TO WAIT UNTIL THE RL11 CONTROLLER IS READY. TO PREVENT HANGING UP
0263 706 : AT HIGH IPL, A MAXIMUM OF 30 USEC ELAPSES BEFORE CONTROL IS
0263 707 : RETURNED TO THE CALLER.
0263 708 :
0263 709 :
7E 50 7D 0263 710 DL_WAIT: ;WAIT FOR CONTROLLER READY
0263 711 MOVQ R0,-(SP) ;SAVE R0, R1
0266 712 DSBINT ;DISABLE INTERRUPTS
026C 713 TIMEWAIT #3,#RL_CS_M_CRDY,RL CS(R4),W
0291 714 ENBINT ;ENABLE INTERRUPTS
50 8E 7D 0294 715 MOVQ (SP)+,R0 ;RESTORE R0, R1
05 0297 716 RSB ;RETURN TO UNIT INIT OR STARTIO
```



```
0298 718 .SBTTL FDT ROUTINE - TEST TRANSFER BYTE COUNT ALIGNMENT
0298 719
0298 720 :++
0298 721
0298 722 DL_ALIGN - FDT ROUTINE TO TEST XFER BYTE COUNT
0298 723
0298 724 FUNCTIONAL DESCRIPTION:
0298 725
0298 726 THIS ROUTINE IS CALLED FROM THE FUNCTION DECISION TABLE DISPATCHER
0298 727 TO CHECK THE BYTE COUNT PARAMETER SPECIFIED BY THE USER PROCESS
0298 728 FOR AN EVEN NUMBER OF BYTES (WORD BOUNDARY).
0298 729
0298 730 INPUTS:
0298 731
0298 732 R3 - IRP ADDRESS (I/O REQUEST PACKET)
0298 733 R4 - PCB ADDRESS (PROCESS CONTROL BLOCK)
0298 734 R5 - UCB ADDRESS (UNIT CONTROL BLOCK)
0298 735 R6 - CCB ADDRESS (CHANNEL CONTROL BLOCK)
0298 736 R7 - BIT NUMBER OF THE I/O FUNCTION CODE
0298 737 R8 - ADDRESS OF FDT TABLE ENTRY FOR THIS ROUTINE
0298 738 4(AP) - ADDRESS OF FIRST FUNCTION DEPENDENT QIO PARAMETER
0298 739
0298 740 OUTPUTS:
0298 741
0298 742 IF THE QIO BYTE COUNT PARAMETER IS ODD, THE I/O OPERATION IS
0298 743 TERMINATED WITH AN ERROR. IF IT IS EVEN, CONTROL IS RETURNED
0298 744 TO THE FDT DISPATCHER.
0298 745
0298 746 :--
0298 747
0298 748 DL_ALIGN:
0298 749 BLBS 4(AP),10$
0298 750 RSB
0298 751 10$: MOVZWL #SS$ IVBUFLN,R0
0298 752 JMP G*EX$ABORTIO
0298 753
0298 754 :CHECK BYTE COUNT AT P1(AP)
0298 755 :IF LBS - ODD BYTE COUNT
0298 756 :EVEN - RETURN TO CALLER
0298 757 :SET BUFFER ALIGNMENT STATUS
0298 758 :ABORT I/O
```

01 04 AC E8  
50 034C 8F 3C  
00000000'GF 17 02A2 752

```
02A8 754 .SBTTL START I/O ROUTINE
02A8 755
02A8 756 :++
02A8 757
02A8 758 DL_STARTIO - START I/O ROUTINE
02A8 759
02A8 760 FUNCTIONAL DESCRIPTION:
02A8 761
02A8 762 THIS FORK PROCESS IS ENTERED FROM THE EXECUTIVE AFTER AN I/O REQUEST
02A8 763 PACKET HAS BEEN DEQUEUED, AND PERFORMS THE FOLLOWING:
02A8 764
02A8 765 - ACTIVATES THE DISK AFTER SETTING UCB FIELDS, OBTAINING
02A8 766 UBA AND CONTROLLER RESOURCES, AND SETTING RL11 REGISTERS
02A8 767
02A8 768 - WAITS FOR AN INTERRUPT
02A8 769
02A8 770 - REGAINS CONTROL AFTER THE ISR SERVICES THE INTERRUPT, AND
02A8 771 - RE-ACTIVATES THE DISK IF THE ORIGINAL FUNCTION
02A8 772 IS NOT YET COMPLETE, OR
02A8 773 - COMPLETES THE I/O REQUEST BY RELEASING RESOURCES,
02A8 774 SETTING STATUS CODES, AND RETURNING TO THE EXECUTIVE.
02A8 775
02A8 776 INPUTS:
02A8 777
02A8 778 R3 - IRP ADDRESS (I/O REQUEST PACKET)
02A8 779 R5 - UCB ADDRESS (UNIT CONTROL BLOCK)
02A8 780 IRP$L_MEDIA - PARAMETER LONGWORD (LOGICAL BLOCK NUMBER)
02A8 781
02A8 782 OUTPUTS:
02A8 783
02A8 784 R0 - FIRST I/O STATUS LONGWORD: STATUS CODE & BYTES XFERED
02A8 785 R1 - SECOND I/O STATUS LONGWORD: 0 FOR DISKS
02A8 786
02A8 787 THE I/O FUNCTION IS EXECUTED.
02A8 788
02A8 789 ALL REGISTERS EXCEPT R0-R4 ARE PRESERVED.
02A8 790
02A8 791 --
02A8 792
02A8 793 DL_STARTIO: ;START I/O OPERATION
02A8 794
02A8 795 :
02A8 796 COMPUTE PHYSICAL MEDIA ADDRESS
02A8 797
02A8 798 LBN = LBN * (SECTORS/BLOCK)
02A8 799 LBN/(SECTORS/TRACK) = D + SECTOR
02A8 800 D/(TRACKS/CYLINDER) = CYLINDER + TRACK
02A8 801
02A8 802
02A8 803 :
02A8 804 PREPROCESS UCB FIELDS
02A8 805
02A8 806
02A8 807 PREPROCESS:
02A8 808 MOVL IRP$L_MEDIA(R3),- ; Copy given MEDIA address (logical)
02A8 809 UCB$L_MEDIA(R5) ; to the UCB.
02AE 810 BBS #IRP$V_PHYSIO,- ;IF SET - PHYSICAL I/O
```



```
50 00BC C5 26 2A A3 02 C5 02B0 811 IRPSW STS(R3),10$
52 44 A5 9A 02B3 812 MULL3 #2,UCBSL_MEDIA(R5),R0 ;SCALE LBN IN R0
00BC C5 50 50 52 7B 02B9 813 MOVZBL UCBSB_SECTORS(R5),R2 ;GET NUMBER OF SECTORS PER TRACK
51 50 50 52 7B 02BD 814 CLRL R1 ;CLEAR HIGH PART OF DIVIDEND
00BD C5 52 45 A5 9A 02BF 815 EDIV R2,R0,R0,UCBSL_MEDIA(R5) ;CALCULATE SECTOR NUMBER AND STORE
00BE C5 50 50 52 7B 02C6 816 MOVZBL UCBSB_TRACKS(R5),R2 ;GET NUMBER OF TRACKS PER CYLINDER
00BD C5 51 90 02CA 817 EDIV R2,R0,R0,R1 ;CALCULATE TRACK AND CYLINDER
00BE C5 50 90 02CF 818 MOVW R1,UCBSL_MEDIA+1(R5) ;STORE TRACK NUMBER
00BD C5 50 80 02D4 819 MOVW R0,UCBSL_MEDIA+2(R5) ;STORE CYLINDER NUMBER
00BD C5 50 80 02D9 820 10$:
0081 C5 90 02D9 821 MOVW UCBSB_ERTMAX(R5),- ;INITIALIZE ERROR RETRY COUNT
0080 C5 02DD 822 UCBSB_ERTCNT(R5)
00C0 C5 7E A5 AE 02E0 823 MNEGW UCBSB_BCNT(R5),UCBSW_BCR(R5) ;INIT NEG BYTES LEFT TO XFER
00D6 C5 B4 02E6 824 CLRW UCBSW_DL_DPN(R5) ;CLEAR DATA PATH NO. FOR USE AS-
00E4 C5 94 02EA 825 ;UBA RESOURCE ALLOCATION FLAG
009A C5 20 A3 80 02EA 826 CLRB UCBSB_DL_DPPE(R5) ;CLEAR DATAPATH PURGE ERROR REGISTER
51 20 A3 00 EF 02EE 827 MOVW IRPSW_FUNC(R3),UCBSW_FUNC(R5) ;SAVE FUNCTION CODE
0092 C5 51 90 02F4 828 EXTZV #IRPSW_FCODE,- ;EXTRACT I/O FUNCTION CODE
51 51 02 91 02F6 829 #IRPSW_FCODE,IRPSW_FUNC(R3),R1 ;...
0092 C5 51 90 02FA 830 MOVW R1,UCBSB_FEX(R5) ;STORE FUNCTION DISPATCH INDEX
51 51 02 91 02FF 831 CMPB #10$_SEER,R1 ;SEEK FUNCTION?
0092 C5 51 12 0302 832 BNEQ 20$ ;IF NEQ - NO
0092 C5 51 12 0304 833 MOVW IRPSL_MEDIA(R3),- ;STORE CYLINDER ADDRESS
00BE C5 51 12 0307 834 UCBSW_DC(R5) ;...
00BE C5 51 12 030A 835 20$:
00BE C5 51 12 030A 836 BICW #UCBSM_DIAGBUF,- ;CLR DIAGNOSTIC BUFFER PRESENT
00BE C5 51 12 030C 837 UCBSW_DEVSTS(R5) ;IF CLR - NO DIAG BUFFER
00BE C5 51 12 030E 838 BBC #IRPSW_DIAGBUF,- ;...
00BE C5 51 12 0310 839 IRPSW_STG(R3),FDISPATCH ;...
00BE C5 51 12 0313 840 BISW #UCBSM_DIAGBUF,UCBSW_DEVSTS(R5) ;SET DIAG BUFFER PRESENT
00BE C5 51 12 0317 841 :
00BE C5 51 12 0317 842 : CENTRAL FUNCTION DISPATCH
00BE C5 51 12 0317 843 :
00BE C5 51 12 0317 844 :
00BE C5 51 12 0317 845 :
00BE C5 51 12 0317 846 FDISPATCH: ;FUNCTION DISPATCH
00BE C5 51 12 0317 847 MOVW UCBSL_IRP(R5),R3 ;GET IRP ADDRESS
00BE C5 51 12 0318 848 BBS #IRPSW_PHYSIO,- ;IF SET - PHYSICAL I/O FUNCTION
00BE C5 51 12 031D 849 IRPSW_STG(R3),10$ ;...
00BE C5 51 12 0320 850 BBS #UCBSW_VALID,- ;IF SET - VOLUME SOFTWARE VALID
00BE C5 51 12 0322 851 UCBSW_STG(R5),10$ ;...
00BE C5 51 12 0325 852 #SS$_VOLINV,R0 ;SET VOLUME INVALID STATUS
00BE C5 51 12 032A 853 BRW RESETXFR ;RESET BYTE COUNT AND EXIT
00BE C5 51 12 032D 854 10$: CLRB UCBSB_DL_DCCHK(R5) ;CLEAR DATA CHECK IN PROGRESS
00BE C5 51 12 0331 855 MOVZBL UCBSB_FEX(R5),R3 ;GET FUNCTION DISPATCH INDEX
00BE C5 51 12 0336 856 CASE R3,- ;DISPATCH TO FUNCTION HANDLING ROUTINE
00BE C5 51 12 0336 857 UNLOAD,- ;UNLOAD
00BE C5 51 12 0336 858 SEEK,- ;SEEK
00BE C5 51 12 0336 859 NOP,- ;RECALIBRATE (unsupported)
00BE C5 51 12 0336 860 DRVCLR,- ;DRVCLR
00BE C5 51 12 0336 861 NOP,- ;RELEASE PORT (unsupported)
00BE C5 51 12 0336 862 NOP,- ;OFFSET HEADS (unsupported)
00BE C5 51 12 0336 863 NOP,- ;RETURN TO CENTER (unsupported)
00BE C5 51 12 0336 864 PACKACK,- ;PACK ACKNOWLEDGE
00BE C5 51 12 0336 865 NOP,- ;SEARCH (unsupported)
00BE C5 51 12 0336 866 WRITECHECK,- ;WRITE CHECK
00BE C5 51 12 0336 867 WRITEDATA,- ;WRITE DATA
```



				0336	868	READDATA,-	: READ DATA
				0336	869	NOP,-	: WRITE HEADER (unsupported)
				0336	870	READHEAD,-	: READ HEADER
				0336	871	NOP,-	: place holder
				0336	872	NOP,-	: place holder
				0336	873	AVAILABLE-	: AVAILABLE
				0336	874	>,LIMIT=#CDF_UNLOAD	:
				035C	875		
				035C	876	NOP:	:NO-OP
				035C	877	SEEK:	:SEEK
				035C	878	DRVCLR:	:DRIVE CLEAR (GET STATUS & RESET)
				035C	879	DO_FUNCTION:	
				035C	880	EXFUNCL RETRYERR	:EXECUTE FUNCTION - RETRY IF FAILURE
	27	11		0360	881	BRB NORMAL	:SUCCESSFUL - EXIT WITH NORMAL STATUS
				0362	882		
				0362	883	PACKACK:	:PACK ACKNOWLEDGE (GET STATUS & RESET)
64	A5	0800	8F	A8	0362	BISW #UCBSM_VALID,-	:Set software volume valid bit.
					0368	UCBSW_STS(R5)	
		F2		11	0368	BRB DO_FUNCTION	:Then go do hardware function.
					036A	887	
					036A	888 UNLOAD:	:UNLOAD
					036A	889 AVAILABLE:	:AVAILABLE
64	A5	0800	8F	AA	036A	890 BICW #UCBSM_VALID,-	:Clear software volume valid bit.
					0370	891 UCBSW_STS(R5)	:and go complete operation without
		17		11	0370	892 BRB NORMAL	:any hardware interaction.
					0372	893	
					0372	894 WRITECHECK:	:WRITE CHECK
					0372	895 READHEAD:	:READ HEADER
	4000	8F		AA	0372	896 BICW #IOSM_DATACHECK,-	:CLEAR DATA CHECK REQUEST-
	009A	C5			0376	897 UCBSW_FUNC(R5)	:TO PREVENT EXTRA WRITE CHECK
					0379	898	
					0379	899 WRITEDATA:	:WRITE DATA
					0379	900 READDATA:	:READ DATA
					0379	901 EXFUNCL RETRYERR,F_SEEK	:EXECUTE EXPLICIT SEEK - RETRY IF FAIL
					0380	902	
53	0092	C5		9A	0380	903 MOVZBL UCBSB_FEX(R5),R3	:GET FUNCTION DISPATCH INDEX
					0385	904 EXFUNCL RETRYERR	:EXECUTE TRANSFER FUNCTION
					0389	905	
					0389	906 :	
					0389	907 :	
					0389	908 :	
					0389	909	
					0389	910 NORMAL:	:SUCCESSFUL OPERATION COMPLETE
	50	01		3C	0389	911 MOVZWL #SS\$ NORMAL,R0	:SET NORMAL COMPLETION STATUS
		0057		31	038C	912 BRW FUNCXT	:FUNCTION EXIT
					038F	913	
					038F	914 RETRYERR:	:RETRIABLE ERROR
	0080	C5		97	038F	915 DECB UCBSB_ERTCNT(R5)	:ANY RETRIES LEFT?
		03		13	0393	916 BEQL FATALERR	:IF EQL - NO
		FF7F		31	0395	917 BRW FDISPATCH	:RETRY FUNCTION
					0398	918	
					0398	919 FATALERR:	:UNRECOVERABLE ERROR
50	0254	8F		3C	0398	920 MOVZWL #SS\$ VOLINV,R0	:ASSUME VOLUME INVALID STATUS
		09		E0	039D	921 BBS #RL_MP_V_VC,-	:IF SET - VOLUME INVALID
	43	00D4	C5		039F	922 UCBSW_DL_MP(R5),FUNCXT	:...
					03A3	923	
50	025C	8F		3C	03A3	924 MOVZWL #SS\$_WRTLCK,R0	:ASSUME WRITE LOCK ERROR STATUS



```
06 00D4 0D E1 03A8 925 BBC #RL_MP_V_WL,- ;IF CLR - VOLUME NOT WRITE LOCKED
00000000 0A E0 03AA 926 UCB$W_DL_MP(R5),5$ ;...
32 00D4 0A E0 03AE 927 BBS #RL_MP_V_WGE,- ;IF SET - WRITE GATE ERROR
00000000 0C 13 03B0 928 UCB$W_DL_MP(R5),FUNCXT ;IF WL & WGE SET - WRITE LOCK ERROR
50 005C 8F 3C 03B4 929 5$: MOVZWL #$$$_DATACHECK,R0 ;ASSUME DATA CHECK ERROR STATUS
00000000 0C 13 03B9 931 TSTB UCB$B_DL_DCHEK(R5) ;WRITE CHECK IN PROGRESS?
00000000 0A E0 03BF 932 BEQL 10$ ;IF EQL - NO
00000000 0A E0 03C1 933 BBS #RL_CS_V_OPI,- ;IF SET - NOT WRITE CHECK ERROR
00000000 0B E0 03C5 934 UCB$W_DL_CS(R5),10$ ;...
1B 00CE 0B E0 03C7 935 BBS #RL_CS_V_CRC,- ;IF SET - WRITE CHECK ERROR
00000000 0C 13 03CB 936 UCB$W_DL_CS(R5),FUNCXT ;...
50 01F4 8F 3C 03CB 938 10$: MOVZWL #$$$_PARITY,R0 ;ASSUME PARITY ERROR STATUS
00000000 0B E0 03D0 939 BBS #RL_CS_V_CRC,- ;IF SET - CRC ERROR
10 00CE 0C 13 03D2 940 UCB$W_DL_CS(R5),FUNCXT ;OR DATAPATH PURGE ERROR
50 008C 8F 3C 03D6 941 20$: MOVZWL #$$$_DRVERR,R0 ;ASSUME DRIVE ERROR STATUS
00000000 0E E0 03DB 943 BBS #RL_CS_V_DE,- ;IF SET - DRIVE ERROR
05 00CE 0C 13 03DD 944 UCB$W_DL_CS(R5),FUNCXT ;...
50 0054 8F 3C 03E1 945 MOVZWL #$$$_CTRLERR,R0 ;ASSUME CONTROLLER ERROR STATUS
00000000 0E E0 03E6 947 50 DD 03E6 948 FUNCXT: ;FUNCTION EXIT
00000000 0A 91 03E8 950 PUSHL R0 ;SAVE FINAL REQUEST STATUS
00000000 0A 91 03EE 951 JSB G*IOC$DIAGBUFILL ;FILL DIAGNOSTIC BUFFER IF PRESENT
00000000 0A 91 03F3 952 CMPB #CDF_WRITECHECK,UCB$B_FEX(R5) ;DRIVE RELATED FUNCTION?
00000000 0A 91 03F5 953 BGTRU 10$ ;IF GTRU - YES
00000000 0A 91 03FA 954 CMPB #CDF_AVAILABLE,UCB$B_FEX(R5) ;DRIVE RELATED FUNCTION?
00000000 0A 91 03FC 955 BEQL 10$ ;IF EQL - YES
53 58 A5 D0 0400 956 MOVL UCB$L_IRP(R5),R3 ;RETRIEVE ADDRESS OF IRP
00000000 0A A1 0404 957 ADDW3 UCB$W_BCR(R5),- ;CALCULATE BYTES TRANSFERRED
02 AE 32 A3 0408 958 TSTW UCB$W_DL_DPN(R5) ;ARE UBA RESOURCES ALLOCATED?
00000000 0A B5 040C 959 BEQL 20$ ;IF EQL - NO
00000000 0A E1 040E 960 BBC #UCB$V_DL_MAPPING,- ;ADAPTER MAPPING?
00000000 0A 13 0410 961 UCB$W_DL_FLAGS(R5),10$ ;IF BC NO
00000000 0A 13 0414 962 RELDPR ;RELEASE DATA PATH
00000000 0A 13 041A 963 RELMPR ;RELEASE MAP REGISTERS
00000000 0A 13 0420 964 BRB 20$ ;JOIN COMMON CODE
00000000 0A 13 0422 965 10$: MOVL UCB$L_DL_SVAPTE(R5),- ;RESTORE ORIGINAL SVAPTE
00000000 0A 13 0426 966 UCB$L_SVAPTE(R5) ;...
00000000 0A 13 0428 967 20$: RELCHAN ;RELEASE CHANNEL IF OWNED
00000000 0A 13 042E 968 CLRL R1 ;CLEAR SECOND STATUS LONGWORD
50 8ED0 0A 13 0430 969 POPL R0 ;RETRIEVE FINAL REQUEST STATUS
00000000 0A 13 0433 971 REQCOM ;COMPLETE REQUEST
```



```
0439 973 :  
0439 974 : FEXL - RL11 HARDWARE FUNCTION EXECUTION  
0439 975 :  
0439 976 : THIS ROUTINE IS CALLED VIA A BSB WITH A BYTE IMMEDIATELY FOLLOWING THAT  
0439 977 : SPECIFIES THE ADDRESS OF AN ERROR ROUTINE. ALL DATA IS ASSUMED TO HAVE BEEN  
0439 978 : SET UP IN THE UCB BEFORE THE CALL. THE APPROPRIATE PARAMETERS ARE LOADED  
0439 979 : INTO DEVICE REGISTERS AND THE FUNCTION IS INITIATED. THE RETURN ADDRESS  
0439 980 : IS STORED IN THE UCB AND A WAITFOR INTERRUPT IS EXECUTED. WHEN THE  
0439 981 : INTERRUPT OCCURS, CONTROL IS RETURNED TO THE CALLER.  
0439 982 :  
0439 983 : INPUTS:  
0439 984 :  
0439 985 : R3 = FUNCTION TABLE DISPATCH INDEX  
0439 986 : R5 = DEVICE UNIT UCB ADDRESS  
0439 987 :  
0439 988 : 00(SP) = RETURN ADDRESS OF CALLER  
0439 989 : 04(SP) = RETURN ADDRESS OF CALLER'S CALLER  
0439 990 :  
0439 991 : IMMEDIATELY FOLLOWING INLINE AT THE CALL SITE IS A BYTE WHICH CONTAINS  
0439 992 : A BRANCH DESTINATION TO AN ERROR RETRY ROUTINE.  
0439 993 :  
0439 994 : OUTPUTS:  
0439 995 :  
0439 996 : THERE ARE FOUR EXITS FROM THIS ROUTINE:  
0439 997 :  
0439 998 : 1. SPECIAL CONDITION - THIS EXIT IS TAKEN IF A POWER FAILURE OCCURS  
0439 999 : OR THE OPERATION TIMES OUT. IT IS A JUMP TO THE APPROPRIATE  
0439 1000 : ERROR ROUTINE.  
0439 1001 :  
0439 1002 : 2. FATAL ERROR - THIS EXIT IS TAKEN IF A FATAL CONTROLLER OR DRIVE  
0439 1003 : ERROR OCCURS OR IF ANY ERROR OCCURS AND ERROR RETRY IS EITHER  
0439 1004 : INHIBITED OR EXHAUSTED. IT IS A JUMP TO THE FATAL ERROR EXIT  
0439 1005 : ROUTINE.  
0439 1006 :  
0439 1007 : 3. RETRIABLE ERROR - THIS EXIT IS TAKEN IF A RETRIABLE CONTROLLER  
0439 1008 : OR DRIVE ERROR OCCURS AND ERROR RETRY IS NEITHER INHIBITED  
0439 1009 : NOR EXHAUSTED. IT CONSISTS OF TAKING THE ERROR BRANCH EXIT  
0439 1010 : SPECIFIED AT THE CALL SITE.  
0439 1011 :  
0439 1012 : 4. SUCCESSFUL OPERATION - THIS EXIT IS TAKEN IF NO ERRORS OCCUR  
0439 1013 : DURING THE OPERATION. IT CONSISTS OF A RETURN INLINE.  
0439 1014 :  
0439 1015 : IN ALL CASES IF AN ERROR OCCURS, AN ATTEMPT IS MADE TO LOG THE ERROR.  
0439 1016 :  
0439 1017 : IN ALL CASES FINAL DEVICE REGISTERS ARE RETURNED VIA THE UCB.  
0439 1018 :  
0439 1019 : UCBSW_BCR(R5) = NEGATIVE BYTES REMAINING TO TRANSFER
```



```
0093 009C C5 8ED0 0439 1021 FEXL:
50 C5 53 90 0439 1022
51 24 A5 D0 043E 1023
04 A1 2C A0 D0 0443 1024
54 55 D1 0447 1025
61 05 12 044B 1026
06 06 11 044F 1027
0451 1028
0454 1029
0456 1030 10$:
045C 1031
045C 1032 20$:
045C 1033
045C 1034
045C 1035
045C 1036
045C 1037
045C 1038
045C 1039
045C 1040
045C 1041
045C 1042
045C 1043
01A3 31 0474 1044 BRW XFER
```

POPL UCBSL\_DPC(R5)  
MOVB R3,UCBSB\_CEX(R5)  
MOVL UCBSL\_CRB(R5),R0  
MOVL CRBSL\_INTD+VECSL\_IDB(R0),R1  
CMPL R5,IDBSL\_OWNER(RT)  
BNEQ 10\$  
MOVL IDBSL\_CSR(R1),R4  
BRB 20\$  
REQPCAN  
CASE R3,<-  
IMMED,-  
IMMED,-  
POSIT,-  
IMMED,-  
DRCLR,-  
IMMED,-  
IMMED,-  
IMMED,-  
DRCLR,-  
IMMED,-  
IMMED,-  
DRCLR,-  
IMMED,-  
>  
XFER

:FUNCTION EXECUTOR  
:SAVE DRIVER PC VALUE  
:SAVE CASE INDEX  
:GET ADDRESS OF PRIMARY CRB  
:GET ADDRESS OF IDB  
:DOES THIS PROCESS OWN CHANNEL?  
:IF NEQ - NO  
:SET ASSIGNED CHANNEL CSR ADDRESS  
:REQUEST CHANNEL (RETURNS R4 = CSR ADR)  
:DISPATCH TO PROPER FUNCTION ROUTINE  
:NO OPERATION  
:UNLOAD VOLUME (NOP)  
:SEEK CYLINDER  
:RECALIBRATE (NOP)  
:DRIVE CLEAR (GET STATUS & RESET)  
:RELEASE DRIVE (NOP)  
:OFFSET HEADS (NOP)  
:RETURN TO CENTERLINE (NOP)  
:PACK ACKNOWLEDGE  
:SEARCH (NOP)  
:TRANSFER FUNCTION

```
0477 1046 :  
0477 1047 : IMMEDIATE FUNCTION EXECUTION  
0477 1048 :  
0477 1049 :     FUNCTIONS INCLUDE:  
0477 1050 :  
0477 1051 :     NO OPERATION,  
0477 1052 :     DRIVE CLEAR, AND  
0477 1053 :     PACK ACKNOWLEDGE  
0477 1054 :  
0477 1055 :     INPUTS:  
0477 1056 :         R3     - CASE INDEX  
0477 1057 :         R4     - CSR ADDRESS  
0477 1058 :         R5     - UCB ADDRESS  
0477 1059 :  
0477 1060 :     FUNCTIONAL DESCRIPTION:  
0477 1061 :  
0477 1062 :     INTERRUPTS ARE LOCKED OUT, THE APPROPRIATE FUNCTION IS INITIATED WITH  
0477 1063 :     INTERRUPT ENABLE, AND A WAITFOR INTERRUPT AND KEEP CHANNEL IS EXECUTED.  
0477 1064 :  
0477 1065 :  
0477 1066 : DRCLR:                                ;DRIVE CLEAR  
0477 1067 :     BISW    #RL_DA_M_STS!-            ;SET GETSTATUS,RESET,AND MARK IN DAR  
0478 1068 :           RL_DA_M_RST!RL_DA_M_MRK,RL_DA(R4) ;...  
0478 1069 :  
0478 1070 : IMMED:                                ;IMMEDIATE FUNCTION EXECUTION  
0478 1071 :     CKPWR                                     ;DISABLE INTERRUPTS, CHECK POWER,-  
0494 1072 :           ;AND PUT UNIT NUMBER IN R2<9:8>  
0494 1073 :     BISW3   R2,FTAB[R3],RL_CS(R4)        ;MERGE UNIT WITH FNTN AND EXECUTE  
0498 1074 :     WFIKPC  RETREG,#2                     ;WAITFOR INTERRUPT  
04A5 1075 :     IOFORK                                     ;RETURN FROM ISR-  
04AB 1076 :           ;CREATE FORK PROCESS (&JSB BACK TO ISR)  
0371 31 04AB 1077 :     BRW     RETREG  
:
```



```
04AE 1079 :  
04AE 1080 : POSITIONING FUNCTION EXECUTION  
04AE 1081 :  
04AE 1082 :     FUNCTIONS INCLUDE:  
04AE 1083 :  
04AE 1084 :         SEEK CYLINDER  
04AE 1085 :  
04AE 1086 :     INPUTS:  
04AE 1087 :         R3      - CASE INDEX  
04AE 1088 :         R4      - DEVICE CSR ADDRESS  
04AE 1089 :         R5      - UCB ADDRESS  
04AE 1090 :  
04AE 1091 :     FUNCTIONAL DESCRIPTION:  
04AE 1092 :  
04AE 1093 :     THE CYLINDER DIFFERENCE WORD IS CALCULATED AND LOADED INTO THE DISK  
04AE 1094 :     ADDRESS REGISTER, INTERRUPTS ARE LOCKED OUT, AND THE SEEK FUNCTION  
04AE 1095 :     IS INITIATED WITHOUT INTERRUPT ENABLE. THE CONTROLLER IS THEN POLLED  
04AE 1096 :     FOR READY, AND DEVICE INTERRUPTS ARE ENABLED.  
04AE 1097 :  
04AE 1098 :     SINCE THE RL01/RL02 DO NOT ISSUE AN INTERRUPT UPON COMPLETION OF A  
04AE 1099 :     SEEK, OVERLAPPED SEEKS ARE NOT ATTEMPTED, AND ONE OF THE FOLLOWING IS  
04AE 1100 :     PERFORMED.  
04AE 1101 :  
04AE 1102 :         IF ONLY A SEEK FUNCTION IS BEING REQUESTED, A DUMMY READ HEADER  
04AE 1103 :         FUNCTION IS ISSUED AND A WAITFOR INTERRUPT IS INITIATED.  
04AE 1104 :         THE READ HEADER IS USED TO SIGNAL THE END OF THE SEEK, SINCE IT  
04AE 1105 :         WILL ISSUE AN INTERRUPT SHORTLY (315 USEC AVG) AFTER THE SEEK IS  
04AE 1106 :         COMPLETE. IT WILL ALSO SENSE FOR A TIMEOUT DURING THE SEEK.  
04AE 1107 :  
04AE 1108 :         IF THE SEEK IS ASSOCIATED WITH A DATA TRANSFER REQUEST (RL01/RL02  
04AE 1109 :         TRANSFER FUNCTIONS REQUIRE EXPLICIT SEEKS), THE PROGRAM KEEPS THE  
04AE 1110 :         CHANNEL AND RETURNS TO FDISPATCH TO ISSUE THE TRANSFER REQUEST  
04AE 1111 :         WHILE THE SEEK IS STILL IN PROGRESS. WHEN THE SEEK COMPLETES, THE  
04AE 1112 :         RL11 CONTROLLER WILL BEGIN THE TRANSFER.  
04AE 1113 :  
04AE 1114 :  
04AE 1115 :     POSIT:                                ;POSITIONING FUNCTION  
04AE 1116 :  
04AE 1117 :     OBTAIN CURRENT DISK ADDRESS  
04AE 1118 :  
04AE 1119 :     IF THERE HAS NOT BEEN A PREVIOUS TRANSFER DURING THIS REQUEST,  
04AE 1120 :     A READ HEADER IS EXECUTED TO DETERMINE THE CURRENT DISK ADDRESS.  
04AE 1121 :  
00D6 C5 B5 04AE 1122 :     TSTW    UCBSW_DL_DPN(R5)      ;WAS THERE A PREVIOUS TRANSFER?  
51 00D2 C5 09 13 04B2 1123 :     BEQL    10$                  ;IF EQL - NO, READ HEADER  
00B5 31 04B4 1124 :     BICW3   #^077,UCBSW_DL_DA(R5),R1 ;PUT CURRENT CYL & SURFACE IN R1  
53 08 9A 04BA 1125 :     BRW     60$                  ;CALCULATE DIFFERENCE WORD  
04BD 1126 :  
04BD 1127 :     MOVZBL  #8,R3                ;SET READ HEADER RETRY COUNT IN R3  
04C0 1128 :     CKPWR   20$                  ;DISABLE INTERRUPTS, CHECK POWER,-  
04D9 1129 :                                ;AND PUT UNIT NUMBER IN R2<9:8>  
0048 8F 52 A9 04D9 1130 :     BISW3   R2,#F_READHEAD!RL_CS_M_1E,- ;EXECUTE READ HEADER  
64 04DE 1131 :                                ;RL CS(R4)  
04DF 1132 :     WFIKPC  40$,#2              ;WAIT FOR INTERRUPT OR TIMEOUT  
04E9 1133 :     IOFORK   ;CREATE FORK PROCESS  
00CE C5 8000 8F B3 04EF 1134 :     BITW    #RL_CS_M_CE,UCBSW_DL_CS(R5) ;ANY ERRORS?  
74 13 04F6 1135 :     BEQL    50$                  ;IF EQL - NO
```



```
53 97 04FB 1136      DECB    R3          ;DECREMENT READ HEADER RETRY COUNT
C4 12 04FA 1137      BNEQ    20$        ;IF NEQ - RETRY READ HEADER
      04FC 1138          ;IF EQL - READ HEADER RETRY EXHAUSTED -
      04FC 1139          ;TRY PREVIOUS TRACK
0081 8F B0 04FC 1140      MOVW    #^0200!RL_DA_M_MRK,- ;LOAD REVERSE SEEK DIFFERENCE WORD
04  A4      0500 1141      RL_DA(R4)
      0502 1142      CKPWR          ;DISABLE INTERRUPTS, CHECK POWER,-
      051B 1143          ;AND PUT UNIT NUMBER IN R2<9:8>
0046 8F 52 A9 051B 1144      BISW3   R2,#F SEEK!RL_CS_M_IE,- ;EXECUTE REVERSE SEEK
64      0520 1145      RL_CSTR4)
      0521 1146      WFIKPC 40$,#2 ;WAIT FOR SEEK TO BEGIN (INTERRUPT)
      052B 1147      IOFORK          ;CREATE FORK PROCESS
      0531 1148      CKPWR          ;DISABLE INTERRUPTS, CHECK POWER,-
      054A 1149          ;AND PUT UNIT NUMBER IN R2<9:8>
0048 8F 52 A9 054A 1150      BISW3   R2,#F READHEAD!RL_CS_M_IE,- ;TRY READ HEADER ON NEW TRACK
64      054F 1151      RL_CSTR4)
      0550 1152      WFIKPC 40$,#2 ;WAITFOR INTERRUPT OR TIMEOUT
00CE C5 8000 8F B3 055A 1153      IOFORK          ;CREATE FORK PROCESS
03      0560 1154      BITW    #RL_CS_M_CE,UCBSW_DL_CS(R5) ;READ HEADER ERROR?
      0567 1155      BEQL    50$        ;IF EQL - NO
      0569 1156      40$:          ;CAN NOT READ CURRENT DISK ADDRESS
      0569 1157      ;          ;CLEAR RETRY COUNT
02B3 31 0569 1158      CLRB    UCBSB_ERTCNT(R5)
      0569 1158      BRW    RETREG
      056C 1159      50$:          ;FOUND CURRENT DISK ADDRESS
51 00D4 C5 3F AB 056C 1160      BICW3   #^077,UCBSW_DL_MP(R5),R1 ;PUT CURRENT CYL & SURFACE IN R1
      0572 1161
      0572 1162      ;
      0572 1163      ; CALCULATE CYLINDER DIFFERENCE WORD
      0572 1164      ;
      0572 1165      ;
50 01 06 00BD 50 D4 0572 1166      60$: CLRL    R0          ;CLEAR R0 FOR DESIRED ADDRESS
50 09 07 00BE C5 F0 0574 1167      INSV    UCBSW_DA+1(R5),#6,#1,R0 ;INSERT DESIRED SURFACE IN R0<6>
      51 50 B1 057B 1168      INSV    UCBSW_DC(R5),#7,#9,R0 ;INSERT DESIRED CYLINDER IN R0<15:7>
      52 13 0582 1169      CMPW    R0,R1 ;IS A SEEK NEEDED?
      51 007F 8F AA 0585 1170      BEQL    80$        ;IF EQL - NO
      50 007F 8F AA 0587 1171      BICW    #^0177,R1 ;REMOVE SURFACE BIT
      51 50 A2 058C 1172      BICW    #^0177,R0 ;REMOVE SURFACE BIT
      08 13 0591 1173      SUBW    R0,R1 ;SUBTRACT DESIRED FROM ACTUAL
      06 1E 0594 1174      BEQL    70$        ;IF EQL - ONLY CHANGE SURFACE
      51 51 AE 0596 1175      BCC     70$        ;IF CC - ACTUAL>=DESIRED
      51 51 A8 0598 1176      MNEGW   R1,R1 ;ACTUAL<DESIRED, MAKE POSITIVE DIFF
51 01 04 00BD C5 F0 059B 1177      BISW    #4,R1 ;SET SIGN FOR MOVE TO CENTER OF DISK
04  A4 51 01 A9 059E 1178      INSV    UCBSW_DA+1(R5),#4,#1,R1 ;INSERT SURFACE BIT
      05A5 1179      BISW3   #RL_DA_M_MRK,R1,RL_DA(R4) ;SET MARKER AND LOAD DIFFERENCE WORD
      05AA 1180
      05AA 1181      ;
      05AA 1182      ; EXECUTE SEEK
      05AA 1183      ;
      05AA 1184      ;
      05AA 1185      CKPWR          ;DISABLE INTERRUPTS, CHECK POWER,-
      05C3 1186          ;AND PUT UNIT NUMBER IN R2<9:8>
0046 8F 52 A9 05C3 1187      BISW3   R2,#F SEEK!RL_CS_M_IE,- ;EXECUTE SEEK FUNCTION
64      05C8 1188      RL_CSTR4)
      05C9 1189      WFIKPC 40$,#2 ;WAIT FOR SEEK TO BEGIN (INTERRUPT)
      05D3 1190      IOFORK          ;CREATE FORK PROCESS
0092 C5 02 91 05D9 1191      80$: CMPB    #10$_SEEK,UCBSB_FEX(R5) ;IS SEEK ASSOCIATED WITH A TRANSFER?
08 13 05DE 1192      BEQL    90$        ;IF EQL - NO, SEEK ONLY
      90$
```



```
05E0 1193
05E0 1194 :
05E0 1195 : RETURN FOR SEEK ASSOCIATED WITH A TRANSFER REQUEST
05E0 1196 :
05E0 1197 :
009C C5 D6 05E0 1198 INCL UCBSL_DPC(R5) ;ADJUST TO CORRECT RETURN ADDRESS
009C D5 17 05E4 1199 JMP @UCBSL_DPC(R5) ;RETURN TO DRIVER FOR TRANSFER
05E8 1200 :
05E8 1201 : RETURN FOR SEEK ONLY REQUEST
05E8 1202 :
05E8 1203 :
05E8 1204 90$: CKPWR ;DISABLE INTERRUPTS, CHECK POWER,-
0601 1205 ;AND PUT UNIT NUMBER IN R2<9:8>
0048 8F 52 A9 0601 1206 BISW3 R2,#F READHEAD!RL_CS_M_IE,- ;EXECUTE DUMMY READ HEADER
64 0606 1207 RL(CSTR4)
0607 1208 WFIKPC RETREG,#2 ;WAIT FOR SEEK TO COMPLETE (INTERRUPT)
0611 1209 IOFORK ;CREATE FORK PROCESS
0205 31 0617 1210 BRW RETREG ;
```

```
061A 1212  
061A 1213 :  
061A 1214 : TRANSFER FUNCTION EXECUTION  
061A 1215 :  
061A 1216 : FUNCTIONS INCLUDE:  
061A 1217 :  
061A 1218 : WRITE CHECK  
061A 1219 : WRITE DATA  
061A 1220 : READ DATA, AND  
061A 1221 : READ HEADER  
061A 1222 :  
061A 1223 : INPUTS:  
061A 1224 : R3 - CASE INDEX  
061A 1225 : R4 - DEVICE CSR ADDRESS  
061A 1226 : R5 - UCB ADDRESS  
061A 1227 :  
061A 1228 : FUNCTIONAL DESCRIPTION:  
061A 1229 :  
061A 1230 : A UNIBUS DATAPATH IS REQUESTED FOLLOWED BY THE APPROPRIATE NUMBER OF MAP  
061A 1231 : REGISTERS REQUIRED FOR THE TRANSFER. THE TRANSFER PARAMETERS ARE LOADED  
061A 1232 : INTO THE DEVICE REGISTERS, INTERRUPTS ARE LOCKED OUT, THE FUNCTION IS  
061A 1233 : INITIATED, AND A WAITFOR INTERRUPT AND KEEP CHANNEL IS EXECUTED.  
061A 1234 :  
061A 1235 : UPON RETURN FROM THE INTERRUPT SERVICE ROUTINE, IF THE TRANSFER IS  
061A 1236 : COMPLETE, THE APPROPRIATE EXIT IS TAKEN. IF THE FUNCTION IS NOT COMPLETE  
061A 1237 : TRANSFER PARAMETERS ARE UPDATED AND A RETURN TO FDISPATCH IS EXECUTED TO  
061A 1238 : RE-ISSUE SEEK AND TRANSFER FUNCTIONS WHILE KEEPING CHANNEL AND UBA  
061A 1239 : RESOURCES. IF A DATA CHECK HAS BEEN REQUESTED, IT IS PERFORMED  
061A 1240 : BEFORE RETURNING TO FDISPATCH.  
061A 1241 :  
061A 1242 :  
061A 1243 : XFER:  
061A 1244 : BBS #UCBSV_DL_MAPPING, - ;TRANSFER FUNCTION EXECUTION  
061C 1245 : UCB$W_DL_FLAGS(R5),2$ ;ADAPTER MAPPING?  
0620 1246 : MOVW UCB$A_DL_BUF_PA(R5),UCB$W_DL_SBA(R5);GET 1ST WORD OF BUFFER ADDR  
0627 1247 : MOVZWL UCB$A_DL_BUF_PA+2(R5),R0;GET BITS 16:21 OF BUFFER ADDRESS  
062C 1248 : MOVW R0,RL_BAE(R4) ;SET MEMORY EXTENSION BITS IN BAE  
0630 1249 : ASHL #4,R0,R0 ;PUT MEMORY EXTENSION BITS IN <5:4>  
0634 1250 : MOVB R0,UCB$B_DL_XBA(R5) ;OF CSR  
0639 1251 :  
0639 1252 : FIRST TRANSFER OF THIS I/O REQUEST - ALLOCATE RESOURCES  
0639 1253 :  
0639 1254 : TSTW UCB$W_DL_DPN(R5) ;RESOURCES ALREADY ALLOCATED?  
063D 1255 : BNEQ 5$ ;IF NEQ - YES  
063F 1256 : CLRL UCB$A_DL_MOVRTN(R5) ;ASSUME READ  
0643 1257 : CMPB #CDF_WRITEDATA,R3 ;WRITE DATA?  
0646 1258 : BNEQ 1$ ;IF NEQ NO  
0648 1259 : MOVAB G^I0CSMOVFRUSER, - ;SET MOVE ROUTINE ADDRESS FOR  
064E 1260 : UCB$A_DL_MOVRTN(R5) ;1ST PARTIAL WRITE  
0651 1261 1$: MOVL UCB$L_SVAPTE(R5),UCB$L_DL_SVAPTE(R5);SAVE SVAPTE FOR BUFFER COPY  
0657 1262 : MNEGW #1,UCB$W_DL_DPN(R5) ;SET FIRST XFER FLAG  
065C 1263 : BRB 5$ ;JOIN COMMON CODE  
065E 1264 :  
065E 1265 :  
065E 1266 : FIRST TRANSFER OF THIS I/O REQUEST - ALLOCATE RESOURCES  
065E 1267 :  
065E 1268 2$: TSTW UCB$W_DL_DPN(R5) ;UBA RESOURCES ALREADY ALLOCATED?
```

00EC C5 3E 00F6 C5 E0 061A 1244  
C5 00F2 C5 B0 061C 1245  
50 00F4 C5 B0 0620 1246  
08 A4 50 B0 0627 1247  
50 50 04 78 062C 1248  
00EB C5 50 90 0630 1249  
0634 1250  
0639 1251  
0639 1252  
0639 1253  
00D6 C5 B5 0639 1254  
61 12 063D 1255  
00E0 C5 D4 063F 1256  
53 08 91 0643 1257  
09 12 0646 1258  
00000000 GF 9E 0648 1259  
00E0 C5 064E 1260  
00D8 C5 78 A5 D0 0651 1261  
00D6 C5 01 AE 0657 1262  
42 11 065C 1263  
065E 1264  
065E 1265  
065E 1266  
065E 1267  
00D6 C5 B5 065E 1268



```
3C 12 0662 1269      BNEQ      5$      ;IF NEQ - YES
      0664 1270      REQDPR      ;REQUEST DATAPATH
      066A 1271      REQMPR      ;REQUEST MAP REGISTERS
      0670 1272      LOADUBA     ;LOAD UNIBUS MAP REGISTERS
51 24 A5 D0 0676 1273      MOVL      UCBSL_CRB(R5),R1      ;GET CRB ADDRESS
      05 00 EF 067A 1274      EXTZV      #VEC$0 DATAPATH,#VEC$5 DATAPATH,- ;EXTRACT DATAPATH NUMBER -
50 37 A1 067D 1275      CRBSL_INTD+VEC$8 DATAPATH(R1),R0 ;FOR UBA RESOURCE FLAG
00D6 C5 50 B0 0680 1276      MOVW      R0,UCBSW_DL_DPN(R5) ;INDICATE UBA RESOURCES ALLOCATED
      0685 1277
50 7C A5 3C 0685 1278      MOVZWL     UCBSW_BOFF(R5),R0 ;GET BYTE OFFSET IN PAGE
      34 A1 F0 0689 1279      INSV      CRBSL_INTD+VEC$W_MAPREG(R1),- ;INSERT HIGH 7 BITS OF ADDRESS
50 07 09 068C 1280      #9,#7,R0
00EC C5 50 B0 068F 1281      MOVW      R0,UCBSW_DL_SBA(R5) ;SET BUFFER ADDRESS
50 34 A1 02 07 EF 0694 1282      EXTZV      #7,#2,CRBSL_INTD+VEC$W_MAPREG(R1),R0 ;GET MEMORY EXTENSION BITS
00EB C5 50 10 85 069A 1283      MULB3     #16,R0,UCBSW_DL_XBA(R5) ;POSITION MEMORY EXTENSION BITS TO <5:4>
      06A0 1284
      06A0 1285      ;
      06A0 1286      ; COMMON TRANSFER POINT
      06A0 1287      ;
      06A0 1288      ;
      06A0 1289      ;
      06A0 1290      ; FOR A READ OPERATION WHEN NO ADAPTER MAPPING IS PRESENT EMPTY THE
      06A0 1291      ; INTERNAL PHYSICALLY CONTIGUOUS BUFFER FROM THE PREVIOUS READ TO THE
      06A0 1292      ; USER'S BUFFER.
      06A0 1293
02DA 30 06A0 1294 5$: BSBW      DL_MOVE_TO_BUFFER      ;COPY TO USER BUFFER
      06A3 1295
      06A3 1296      ; PUT BUFFER ADDRESS, WORD COUNT, AND DISK ADDRESS IN DEVICE REGISTERS
      06A3 1297      ;
      06A3 1298
02 A4 00EC C5 B0 06A3 1299      MOVW      UCBSW_DL_SBA(R5),RL_BA(R4) ;SET BUFFER ADDRESS
      00C0 C5 AE 06A9 1300      MNEGW     UCBSW_BCR(R5),- ;GET BYTES LEFT TO TRANSFER AND -
      00CC C5 06AD 1301      UCBSW_DL_PBCR(R5) ;ASSUME ONLY ONE TRANSFER NEEDED
52 44 A5 9A 06B0 1302      MOVZBL     UCBSW_SECTORS(R5),R2 ;GET SECTORS/SURFACE
51 00BC C5 9A 06B4 1303      MOVZBL     UCBSW_DA(R5),R1 ;GET DESIRED SECTOR
      52 51 A2 06B9 1304      SUBW      R1,R2 ;CALCULATE SECTORS LEFT ON SURFACE
52 0100 8F A4 06BC 1305      MULW      #256,R2 ;CONVERT TO BYTES LEFT ON SURFACE
52 00CC C5 B1 06C1 1306      CMPW      UCBSW_DL_PBCR(R5),R2 ;ARE ADDITIONAL TRANSFERS REQUIRED?
      05 1B 06C6 1307      BLEQU     10$ ;IF LEQU - NO
00CC C5 52 B0 06C8 1308      MOVW      R2,UCBSW_DL_PBCR(R5) ;SET BYTE COUNT FOR THIS TRANSFER
      06CD 1309
      06CD 1310      ; FOR A WRITE OPERATION WHEN NO ADAPTER MAPPING IS PRESENT
      06CD 1311      ; FILL INTERNAL PHYSICALLY CONTIGUOUS BUFFER FROM THE USER'S BUFFER.
      06CD 1312
02F2 30 06CD 1313 10$: BSBW      DL_MOVE_FROM_BUFFER      ;COPY FROM USER BUFFER
      06D0 1314
50 00EB C5 9A 06D0 1315      MOVZBL     UCBSW_DL_XBA(R5),R0 ;SET MEMORY EXTENSION BITS
50 F95E CF43 AB 06D5 1316      BISW      FTAB[R3],R0 ;MERGE XBA BITS WITH FUNCTION
52 00CC C5 02 A7 06DB 1317      DIVW3     #2,UCBSW_DL_PBCR(R5),R2 ;CALCULATE TRANSFER WORD COUNT
      06 A4 52 AE 06E1 1318      MNEGW     R2,RL_MPTR4 ;SET TRANSFER WORD COUNT
      06E5 1319
51 01 51 00BC C5 9A 06E5 1320      MOVZBL     UCBSW_DA(R5),R1 ;PUT DESIRED SECTOR IN R1<5:0>
51 09 06 00BD C5 F0 06EA 1321      INSV      UCBSW_DA+1(R5),#6,#1,R1 ;INSERT DESIRED SURFACE IN R1<6>
      07 00BE C5 F0 06F1 1322      INSV      UCBSW_DC(R5),#7,#9,R1 ;INSERT DESIRED CYLINDER IN R1<15:7>
      04 A4 51 B0 06F8 1323      MOVW      R1,RL_DA(R4) ;SET DESIRED DISK ADDRESS
      06FC 1324
      06FC 1325 ;
```



```
06FC 1326 : EXECUTE THE TRANSFER FUNCTION
06FC 1327 :
06FC 1328 :
06FC 1329 CKPWR :DISABLE INTERRUPTS, CHECK POWER,-
0715 1330 :AND PUT UNIT NUMBER IN R2<9:8>
64 50 52 A9 0715 1331 BISW3 R2,R0,RL_CS(R4) :EXECUTE FUNCTION
0719 1332 WFIKPC RETREG,#8 :WAITFOR INTERRUPT AND KEEP CHANNEL
0723 1333 :RETURN HERE FROM ISR SAVING REGISTERS
0723 1334 IOFORK :CREATE FORK PROCESS (RETURN TO ISR)
0729 1335 :RETURN HERE FROM ISR REI ROUTINE
0729 1336 :
0729 1337 :
0729 1338 : PURGE DATAPATH
0729 1339 :
0729 1340 :
00E4 C5 94 0729 1341 CLRB UCBSB_DL_DPPE(R5) :CLEAR DATAPATH PURGE ERROR
00000000 GF 16 072D 1342 JSB G*IOCSPURGDATAP :PURGE DATAPATH
04 50 E8 0733 1343 BLBS R0,20$ :IF SET - NO PURGE ERRORS
00E4 C5 96 0736 1344 INCB UCBSB_DL_DPPE(R5) :SET DATAPATH PURGE ERROR
073A 1345 :
073A 1346 :
073A 1347 : SAVE UBA REGISTERS FOR UPDATE AND REGDUMP ROUTINES
073A 1348 :
073A 1349 :
50 00D0 C5 01 E1 073A 1350 20$: BBC #UCBSV_DL_MAPPING,- :ADAPTER MAPPING?
51 00DB C5 51 DO 073C 1351 UCBSW_DL_FLAGS(R5),30$ :IF BC NO
00CE C5 07 09 EF 0740 1352 MOVL R1,UCBSL_DL_DPR(R5) :SAVE DATAPATH REGISTER
50 02 07 51 FO 0745 1353 EXTZV #9,#7,UCBSW_DL_BA(R5),R0 :EXTRACT LOW BITS OF FINAL MAP REG NO.
50 01EF 8F B1 074C 1354 EXTZV #4,#2,UCBSW_DL_CS(R5),R1 :EXTRACT HI BITS OF FINAL MAP REG NO.
50 05 18 0753 1355 INSV R1,#7,#2,R0 :INSERT HIGH BITS OF FINAL MAP REGISTER
01EF 8F 3C 0758 1356 CMPW #495,R0 :LEGAL MAP REGISTER NUMBER?
50 05 18 075D 1357 BGEQ 25$ :IF GEQ - YES
01EF 8F 3C 075F 1358 MOVZWL #495,R0 :RESTRICT MAP REGISTER NUMBER
00DC C5 6240 DO 0764 1359 25$: MOVL (R2)[R0],UCBSL_DL_FMPR(R5) :SAVE FINAL MAP REGISTER NUMBER
00E0 C5 D4 076A 1360 CLRL UCBSL_DL_PMPR(R5) :CLEAR PREVIOUS MAP REGISTER CONTENTS
50 D7 076E 1361 DECL R0 :CALCULATE PREVIOUS MAP REGISTER NUMBER
0F 00 EC 0770 1362 CMPV #VECSV_MAPREG,#VECS MAPREG,- :ANY PREVIOUS MAP REGISTER?
50 34 A3 0773 1363 CRBSL_INTD+VECSW_MAPREG(R3),R0 :...
00E0 C5 6240 DO 0776 1364 BGTR 30$ :IF GTR - NO
03 00CE C5 0F E1 0778 1365 MOVL (R2)[R0],UCBSL_DL_PMPR(R5) :SAVE PREVIOUS MAP REGISTER
0098 31 077E 1366 30$: BBC #RL_CS_V_CE,UCBSW_DL_CS(R5),40$ :IF CLR - NO RL ERRORS
03 00E4 C5 E9 0784 1367 BRW RETREG :DEVICE ERROR
0090 31 0787 1368 40$: BLBC UCBSB_DL_DPPE(R5),45$ :IF CLR - NO PURGE ERROR
078C 1369 BRW RETREG :PURGE ERROR
078F 1370 :
078F 1371 :
078F 1372 : RETURN HEADER INFORMATION FOR READ HEADER FUNCTION
078F 1373 :
078F 1374 :
0093 C5 0E 91 078F 1375 45$: CMPB #CDF_READHEAD,UCBSB_CEX(R5) :READ HEADER FUNCTION?
2F 12 0794 1376 BNEQ DATAHECK :IF NEQ - NO
00C0 C5 DD 0796 1377 PUSHL UCBSW_BCR(R5) :SAVE NEG BYTES REMAINING
78 A5 DD 079A 1378 PUSHL UCBSL_SVAPTE(R5) :SAVE ADDRESS OF PTE
51 00E5 C5 9E 079D 1379 MOVAB UCBSW_DL_DB(R5),R1 :SET ADDRESS OF INTERNAL BUFFER
52 06 DO 07A2 1380 MOVL #6,R2 :SET NUMBER OF BYTES TO MOVE
7E A5 52 B1 07A5 1381 CMPW R2,UCBSW_BCNT(R5) :ROOM FOR FULL HEADER?
04 1F 07A9 1382 BLSSU 50$ :IF LSSU - YES
```



```
00C0 C5 52 7E A5 3C 07AB 1383 MOVZWL UCBSW_BCNT(R5),R2 ;SET LENGTH OF PARTIAL HEADER
52 7E A5 A3 07AF 1384 50$: SUBW3 UCBSW_BCNT(R5),R2,UCBSW_BCR(R5) ;CALCULATE TRANSFER BYTE COUNT
00000000 GF 16 07B6 1385 JSB G*IOCSMOVTOUSER ;MOVE HEADER TO USER BUFFER
78 A5 8ED0 07BC 1386 POPL UCBSL_SVAPTE(R5) ;RESTORE ADDRESS OF PTE
00C0 C5 8ED0 07C0 1387 POPL UCBSW_BCR(R5) ;RESTORE NEG BYTES REMAINING
07C5 1388
07C5 1389
07C5 1390 : PERFORM DATA CHECK, IF REQUESTED
07C5 1391
07C5 1392
07C5 1393 DATACHECK: ;DATACHECK AFTER PARTIAL TRANSFER
10 009A OE E1 07C5 1394 BBC #IOSV_DATACHECK,- ;IF CLR - DATA CHECK NOT REQUESTED
00C9 C5 00 07C7 1395 UCBSW_FUNC(R5),UPDATE ;
0A E4 07CB 1396 BBSC #0,UCBSB_DL_DCHEK(R5),- ;IF SET - DATA CHECK ALREADY PERFORMED
00C9 C5 96 07D1 1397 UPDATE ;
53 0A 9A 07D5 1398 INCB UCBSB_DL_DCHEK(R5) ;SET DATA CHECK IN PROGRESS
FE3F 31 07D8 1400 MOVZBL #IOS_WRITECHECK,R3 ;SET CASE INDEX TO WRITE CHECK
07DB 1401 BRW XFER ;BRANCH TO PERFORM WRITE CHECK
07DB 1402
07DB 1403 : UPDATE BUFFER ADDRESS, CURRENT DISK ADDRESS, AND BYTES REMAINING
07DB 1404 : FOR NEXT TRANSFER
07DB 1405
07DB 1406
07DB 1407 UPDATE: ;UPDATE TRANSFER PARAMETERS
10 00F6 01 E1 07DB 1408 BBC #UCBSV_DL_MAPPING,- ;ADAPTER MAPPING?
00CE C5 CF 8F 8B 07DD 1409 UCBSW_DL_FLAGS(R5),10$ ;IF BC NO
00EB C5 07E1 1410 BICB3 #*XCF,UCBSW_DL_CS(R5),- ;SAVE MEMORY EXTENSION BITS
00D0 C5 B0 07E7 1411 UCBSB_DL_XBA(R5) ;
00EC C5 07EA 1412 MOVW UCBSW_DL_BA(R5),- ;UPDATE SAVED BUFFER ADDRESS
07EE 1413 UCBSW_DL_SBA(R5) ;...
07F1 1414
00BC C5 94 07F1 1415 10$: CLRB UCBSW_DA(R5) ;UPDATE DESIRED SECTOR TO ZERO
51 00D2 C5 00000040 8F C1 07F5 1416 ADDL3 #*0100,UCBSW_DL_DA(R5),R1 ;INCREMENT CYLINDER & SURFACE
52 51 01 06 EF 07FF 1417 EXTZV #6,#1,R1,R2 ;EXTRACT DESIRED DISK SURFACE
00BD C5 52 90 0804 1418 MOVB R2,UCBSW_DA+1(R5) ;UPDATE DESIRED DISK SURFACE
52 51 09 07 EF 0809 1419 EXTZV #7,#9,R1,R2 ;EXTRACT DESIRED DISK CYLINDER
00BE C5 52 B0 080E 1420 MOVW R2,UCBSW_DC(R5) ;UPDATE DESIRED DISK CYLINDER
00CC C5 A0 0813 1421 ADDW UCBSW_DL_PBCR(R5),- ;UPDATE NEG BYTES REMAINING TO XFER
00C0 C5 0817 1422 UCBSW_BCR(R5) ;
03 13 081A 1423 BEQL RETREG ;IF EQL - TRANSFER COMPLETE
FAF8 31 081C 1424 BRW FDISPATCH ;MORE BYTES REMAINING - CONTINUE
081F 1425
081F 1426 : GET STATUS AND RESET ERRORS
081F 1427
081F 1428
081F 1429
081F 1430 RETREG: ;GET STATUS AND RESET ERRORS
081F 1431
081F 1432 : FOR A READ OPERATION WHEN NO ADAPTER MAPPING IS PRESENT
081F 1433 : EMPTY INTERNAL BUFFER INTO USER'S BUFFER FOR LAST READ
081F 1434
015B 30 081F 1435 BSBW DL_MOVE_TO_BUFFER ;MOVE LAST READ INTO USER'S BUFFER
0822 1436
0822 1437 SETIPL UCBSB_FIPL(R5) ;MAKE SURE AT FORK IPL (TIMEOUT)
04 A4 03 B0 0826 1438 MOVW #RL_DA_M_STS,- ;PUT GET STATUS IN DAR
082A 1439 RL_DA_M_MRK,RL_DA(R4) ;...
```



```
52 08 08 54 52 D4 082A 1440 CLRL R2 ;CLEAR R2 FOR UNIT NUMBER
    64 04 52 A5 F0 082C 1441 INSV UCBSW_UNIT(R5),#8,#8,R2 ;GET UNIT NUMBER
    FA2A 30 0832 1442 BISW3 R2,#F-GETSTATUS,RL_CS(R4) ;EXECUTE GET STATUS
    00D4 C5 06 A4 B0 0836 1443 BSBW DL_WAIT ;WAIT FOR CONTROLLER
    04 A4 0B B0 0839 1444 MOVW RL_MP(R4),UCBSW_DL_MP(R5) ;RETRIEVE ERROR REGISTER
    64 04 52 A9 083F 1445 MOVW #RL_DA_M_RST!- ;PUT GET STATUS & RESET IN DAR
    FA19 30 0843 1446 RL_DA_M_STS!RL_DA_M_MRK,RL_DA(R4) ;...
    0843 1447 BISW3 R2,#F-GETSTATUS,RL_CS(R4) ;EXECUTE RESET
    0844 1448 BSBW DL_WAIT ;WAIT FOR CONTROLLER
    084A 1449
    084A 1450 ;
    084A 1451 ; DETERMINE EXIT - SPECIAL CONDITION, FATAL ERROR, RETRIABLE ERROR, OR SUCCESS
    084A 1452 ;
    084A 1453 ;
    00D4 C5 05 00 ED 084A 1454 CMPZV #0,#5,UCBSW_DL_MP(R5),- ;HEADS, BRUSHES, STATE OK?
    1D 0850 1455 #RL_MP_M_BH!RL_MP_M_HO!RL_SLM ;...
    OE 13 0851 1456 BEQL 1$ ;IF EQL - YES, ONLINE
    64 A5 0040 8F AA 0853 1457 BICW #UCBSM_TIMEOUT,UCBSW_STS(R5) ;CLEAR DEVICE TIME OUT
    50 01A4 8F 3C 0859 1458 MOVZWL #SS$_MEDOFFL,R0 ;SET MEDIUM OFFLINE STATUS
    FB85 31 085E 1459 BRW FUNCRT ;RETURN
    64 A5 0060 8F B3 0861 1460 1$: BITW #UCBSM_POWER!- ;POWER FAIL OR DEVICE TIMEOUT?
    0867 1461 UCBSM_TIMEOUT,UCBSW_STS(R5) ;...
    53 12 0867 1462 BNEQ SPECOND ;IF NEQ - YES, SPECIAL CONDITION
    0869 1463
    4A 00D4 C5 09 E0 0869 1464 BBS #RL_MP_V_VC,UCBSW_DL_MP(R5),20$ ;IF SET - VOLUME INVALID
    05 00CE C5 0F E0 086F 1465 BBS #RL_CS_V_CE,UCBSW_DL_CS(R5),2$ ;IF SET - RL ERROR
    37 00E4 C5 E9 0875 1466 BLBC UCBSB_DL_DPPE(R5),10$ ;IF CLR - NO PURGE ERROR
    00000000 GF 16 087A 1467 2$: JSB G*ERL$DEVICERR ;ALLOCATE AND FILL ERROR MESSAGE BUFFER
    33 009A C5 0F E0 0880 1468 BBS #IOSV_INHRETRY,UCBSW_FUNC(R5),20$ ;IF SET - RETRY INHIBITED
    2D 00CE C5 0D E0 0886 1469 BBS #RL_CS_V_NXM,UCBSW_DC_CS(R5),20$ ;IF SET - NONEXISTENT MEMORY
    15 00CE C5 0E E1 088C 1470 BBC #RL_CS_V_DE,UCBSW_DL_CS(R5),5$ ;IF CLR - NO DRIVE ERRORS
    06 00D4 C5 0D E1 0892 1471 BBC #RL_MP_V_WL,UCBSW_DL_MP(R5),4$ ;IF CLR - NOT WRITE LOCKED
    1B 00D4 C5 0A E0 0898 1472 BBS #RL_MP_V_WGE,UCBSW_DC_MP(R5),20$ ;IF WL & WGE SET - WL ERROR
    00D4 C5 C500 8F B3 089E 1473 4$: BITW #RL_MP_M_WDE!- ;WRITE DATA ERROR, OR
    08A5 1474 RL_MP_M_CHE!- ;CURRENT HEAD ERROR, OR
    08A5 1475 RL_MP_M_WGE!- ;WRITE GATE ERROR, OR
    08A5 1476 RL_MP_M_DSE,UCBSW_DL_MP(R5) ;DRIVE SELECT ERROR?
    12 12 08A5 1477 BNEQ 20$ ;IF NEQ - YES
    08A7 1478
    08A7 1479 ;
    08A7 1480 ; RETRIABLE ERROR EXIT
    08A7 1481 ;
    08A7 1482 ;
    7E 009C D5 98 08A7 1483 5$: CVTBL @UCBSL_DPC(R5),-(SP) ;GET BRANCH DISPLACEMENT
    009C C5 8E C0 08AC 1484 ADDL (SP)+,UCBSL_DPC(R5) ;CALCULATE RETURN ADDRESS - 1
    08B1 1485
    08B1 1486 ;
    08B1 1487 ; SUCCESSFUL OPERATION EXIT
    08B1 1488 ;
    08B1 1489 ;
    009C C5 D6 08B1 1490 10$: INCL UCBSL_DPC(R5) ;ADJUST TO CORRECT RETURN ADDRESS
    009C D5 17 08B5 1491 JMP @UCBSL_DPC(R5) ;RETURN TO DRIVER
    08B9 1492
    08B9 1493 ;
    08B9 1494 ; FATAL ERROR EXIT
    08B9 1495 ;
    08B9 1496 ;
```



```
FADC 31 08B9 1497 20$: BRW FATALERR ;FATAL ERROR EXIT
      08BC 1498
      08BC 1499 ;
      08BC 1500 ; SPECIAL CONDITION EXIT (POWER FAILURE OR DEVICE TIMEOUT)
      08BC 1501 ;
      08BC 1502
      08BC 1503 SPECOND:
27 64 A5 05 E0 08BC 1504 BBS #UCBSV_POWER,UCBSW_STS(R5),PWRFAIL ;IF SET - POWER FAILURE
      08C1 1505 ;IF CLR - DEVICE TIMEOUT
      08C1 1506 JSB G^ERL$DEVICTMO ;LOG DEVICE TIMEOUT
64 A5 0040 8F 16 08C1 1506 JSB G^ERL$DEVICTMO ;LOG DEVICE TIMEOUT
50 022C 8F AA 08C7 1507 BICW #UCBSM_TIMEOUT,UCBSW_STS(R5) ;CLEAR TIMEOUT STATUS
      0080 C5 3C 08CD 1508 MOVZWL #SS$ TIMEOUT,R0 ;SET DEVICE TIMEOUT STATUS
      03 13 08D2 1509 DECB UCBSB_ERTCNT(R5) ;ANY ERROR RETRIES REMAINING?
      FA3C 31 08D6 1510 BEQL RESETXFR ;IF EQL - NO
      08DB 1511 BRW FDISPATCH ;RETURN
      08DB 1512
      08DB 1513 RESETXFR: ;RESET TRANSFER BYTE COUNT
53 58 A5 D0 08DB 1514 MOVL UCBSL_IRP(R5),R3 ;GET ADDRESS OF I/O PACKET
00C0 C5 32 A3 AE 08DF 1515 MNEGW IRPSW_BCNT(R3),UCBSW_BCR(R5) ;RESET BYTE COUNT
      FAFE 31 08E5 1516 BRW FUNCXT ;EXIT
      08E8 1517
      08E8 1518 PWRFAIL: ;POWER FAILURE
64 A5 20 AA 08E8 1519 BICW #UCBSM_POWER,UCBSW_STS(R5) ;CLEAR POWER FAILURE BIT
      00D6 C5 B5 08EC 1520 TSTW UCBSW_DL_DPN(R5) ;ARE UCB RESOURCES ALLOCATED?
      12 13 08F0 1521 BEQL 50$ ;IF EQL - NO
      01 E1 08F2 1522 BBC #UCBSV_DL_MAPPING,- ;ADAPTER MAPPING?
OC 00F6 C5 08F4 1523 UCBSW_DL_FLAGS(R5),50$ ;IF BC NO
      08F8 1524 RELDPR ;RELEASE DATA PATH
      08FE 1525 RELMPR ;RELEASE MAP REGISTERS
53 58 A5 D0 0904 1526 50$: RELCHAN ;RELEASE CHANNEL IF OWNED
      2C A3 7D 090A 1527 MOVL UCBSL_IRP(R5),R3 ;GET ADDRESS OF I/O PACKET
      78 A5 7D 090E 1528 MOVQ IRPSL_SVAPTE(R3),- ;RESTORE TRANSFER PARAMETERS
      F992 31 0911 1529 UCBSL_SVAPTE(R5) ;
      0913 1530 BRW PREPROCESS ;RETURN TO PREPROCESS UCB FIELDS
```



```
0916 1532 .SBTTL INTERRUPT SERVICE ROUTINE
0916 1533 :++
0916 1534 : DL$INT - RL11 INTERRUPT SERVICE ROUTINE
0916 1535 :
0916 1536 : FUNCTIONAL DESCRIPTION:
0916 1537 :
0916 1538 : THIS ROUTINE IS ENTERED VIA A JSB INSTRUCTION WHEN AN INTERRUPT
0916 1539 : OCCURS ON AN RL11 DISK CONTROLLER. IF THE INTERRUPT IS NOT EXPECTED,
0916 1540 : THE UNSOLICITED INTERRUPT ROUTINE DISMISSES THE INTERRUPT. IF
0916 1541 : THE INTERRUPT IS EXPECTED, DEVICE REGISTERS ARE SAVED AND THE
0916 1542 : DRIVER IS CALLED AT ITS INTERRUPT RETURN ADDRESS. THE DRIVER FORKS,
0916 1543 : CAUSING A RETURN TO THIS ROUTINE, WHICH RESTORES GENERAL REGISTERS
0916 1544 : AND DISMISSES THE INTERRUPT.
0916 1545 :
0916 1546 : INPUTS:
0916 1547 :
0916 1548 : 00(SP) - POINTER TO ADDRESS OF THE IDB
0916 1549 : 04(SP) - SAVED R0
0916 1550 : 08(SP) - SAVED R1
0916 1551 : 12(SP) - SAVED R2
0916 1552 : 16(SP) - SAVED R3
0916 1553 : 20(SP) - SAVED R4
0916 1554 : 24(SP) - SAVED R5
0916 1555 : 28(SP) - PC AT THE TIME OF THE INTERRUPT
0916 1556 : 32(SP) - PSL AT THE TIME OF THE INTERRUPT
0916 1557 :
0916 1558 : OUTPUTS:
0916 1559 :
0916 1560 : DEVICE REGISTERS ARE SAVED, IPL IS LOWERED TO FORK LEVEL, THE
0916 1561 : INTERRUPT IS DISMISSED, ALL REGISTERS EXCEPT R0-R5 ARE PRESERVED.
0916 1562 :
0916 1563 :--
0916 1564 :
0916 1565 DL_INT:: : INTERRUPT SERVICE ROUTINE
53 9E D0 0916 1566 MOVL @ (SP)+, R3 : REMOVE ADDRESS OF IDB FROM STACK
54 63 7D 0919 1567 MOVQ (R3), R4 : GET ADDRESS OF CSR AND UCB
55 55 D5 091C 1568 TSTL R5 : IS R5 A ZERO
39 13 01 E5 091E 1569 BEQL DL_UNSOINT : IF EQL NO OWNER
01 01 E5 0920 1570 BBCC #UCB$V_INT, - : IF CLR - INTERRUPT NOT EXPECTED
34 64 A5 0922 1571 UCB$W_STS(R5), DL_UNSOINT : ...
0925 1572 :
0093 C5 0E 91 0925 1573 CMPB #CDF_READHEAD, UCB$B_CEX(R5) : READ HEADER FUNCTION?
12 12 092A 1574 BNEQ 10$ : IF NEQ - NO
00E5 C5 06 A4 B0 092C 1575 MOVW RL_MP(R4), UCB$W_DL_DB(R5) : SAVE SECTOR HEADER INFORMATION
00E7 C5 06 A4 B0 0932 1576 MOVW RL_MP(R4), UCB$W_DL_DB+2(R5) : ...
00E9 C5 06 A4 B0 0938 1577 MOVW RL_MP(R4), UCB$W_DL_DB+4(R5) : ...
093E 1578 :
53 52 64 9E 093E 1579 10$: MOVAB RL_CS(R4), R2 : GET ADDRESS OF CONTROL STATUS REGISTER
00CE C5 9E 0941 1580 MOVAB UCB$W_DL_CS(R5), R3 : GET ADDRESS OF REGISTER SAVE AREA
83 82 B0 0946 1581 MOVW (R2)+, (R3)+ : SAVE CONTROL STATUS REGISTER
83 82 B0 0949 1582 MOVW (R2)+, (R3)+ : SAVE BUFFER ADDRESS REGISTER
83 82 B0 094C 1583 MOVW (R2)+, (R3)+ : SAVE DISK ADDRESS REGISTER
83 82 B0 094F 1584 MOVW (R2)+, (R3)+ : SAVE MULTIPURPOSE REGISTER
0952 1585 :
53 10 A5 7D 0952 1586 20$: MOVQ UCB$L_FR3(R5), R3 : RESTORE DRIVER CONTEXT
OC B5 16 0956 1587 JSB @UCB$L_FPC(R5) : CALL DRIVER AT INTERRUPT RETURN ADDRESS
0959 1588
```



DLDRIVER  
V04-000

- VAX/VMS RL11/RL01,RL02 DISK DRIVER<sup>D 8</sup>  
INTERRUPT SERVICE ROUTINE

16-SEP-1984 00:17:29 VAX/VMS Macro V04-00  
5-SEP-1984 00:12:24 [DRIVER.SRC]DLDRIVER.MAR;1

Page 36  
(1)

3F	BA	0959	1589	DL_UNSolNT:			
	02	0959	1590	POPR	#^M<R0,R1,R2,R3,R4,R5>	:UNSOLICITED INTERRUPT	
		095B	1591	REI		:RESTORE R0-R5	
						:RETURN FROM INTERRUPT	

DMD  
V04



```
095C 1593 .SBTTL REGISTER DUMP ROUTINE
095C 1594 :++
095C 1595 :
095C 1596 : DL_REGDUMP - REGISTER DUMP ROUTINE
095C 1597 :
095C 1598 : FUNCTIONAL DESCRIPTION:
095C 1599 :
095C 1600 : THIS ROUTINE IS CALLED TO SAVE THE DEVICE REGISTERS AND UBA RESOURCE
095C 1601 : REGISTERS IN A SPECIFIED BUFFER. IT IS CALLED FROM THE DEVICE ERROR
095C 1602 : LOGGING ROUTINE AND FROM THE DIAGNOSTIC BUFFER FILL ROUTINE.
095C 1603 :
095C 1604 : INPUTS:
095C 1605 :
095C 1606 : R0 - ADDRESS OF REGISTER SAVE BUFFER
095C 1607 : R4 - ADDRESS OF DEVICE CONTROL STATUS REGISTER (CSR)
095C 1608 : R5 - ADDRESS OF UNIT CONTROL BLOCK (UCB)
095C 1609 :
095C 1610 : OUTPUTS:
095C 1611 :
095C 1612 : THE DEVICE AND UBA REGISTERS ARE SAVED IN THE SPECIFIED BUFFER.
095C 1613 : R0 CONTAINS THE ADDRESS OF THE NEXT EMPTY LONGWORD IN THE BUFFER.
095C 1614 : ALL REGISTERS EXCEPT R1 AND R2 ARE PRESERVED.
095C 1615 :
095C 1616 :--
095C 1617 :
095C 1618 DL_REGDUMP:
095C 1619 MOVL #<RL_NUM_REGS+5>,(R0)+
095C 1620 MOVAL UCBSW DL_CS(R5),R1
095C 1621 MOVZBL #RL_NUM_REGS,R2
095C 1622 10$: MOVZWL (R1)+,(R0)+
095C 1623 SOBGTR R2,10$
095C 1624 MOVZWL (R1)+,(R0)+
095C 1625 MOVL (R1)+,(R0)+
095C 1626 MOVL (R1)+,(R0)+
095C 1627 MOVL (R1)+,(R0)+
095C 1628 MOVZBL (R1)+,(R0)+
095C 1629 RSB
095C 1630

51 80 09 D0 095C 1619 MOVL #<RL_NUM_REGS+5>,(R0)+
    00CE C5 DE 095F 1620 MOVAL UCBSW DL_CS(R5),R1
    52 04 9A 0964 1621 MOVZBL #RL_NUM_REGS,R2
    80 81 3C 0967 1622 10$: MOVZWL (R1)+,(R0)+
    FA 52 F5 096A 1623 SOBGTR R2,10$
    80 81 3C 096D 1624 MOVZWL (R1)+,(R0)+
    80 81 D0 0970 1625 MOVL (R1)+,(R0)+
    80 81 D0 0973 1626 MOVL (R1)+,(R0)+
    80 81 D0 0976 1627 MOVL (R1)+,(R0)+
    80 81 9A 0979 1628 MOVZBL (R1)+,(R0)+
    05 097C 1629 RSB
    097D 1630
```

```
:REGISTER DUMP ROUTINE
:INSERT NUMBER OF REGISTERS
:GET ADDRESS OF SAVED DEVICE REGISTERS
:GET NUMBER OF DEVICE REGISTERS TO MOVE
:DUMP REGISTER IN BUFFER
:IF GTR - STILL MORE TO MOVE
:DUMP DATAPATH NUMBER
:DUMP DATAPATH REGISTER
:DUMP FINAL MAP REGISTER
:DUMP PREVIOUS MAP REGISTER
:DUMP DATAPATH PURGE ERROR REGISTER
:RETURN
```



```
097D 1632 .SBTTL MOVE TO USER BUFFER ROUTINE
097D 1633 :++
097D 1634 :
097D 1635 : DL_MOVE_TO_BUFFER - MOVE TO USER BUFFER
097D 1636 :
097D 1637 : FUNCTIONAL DESCRIPTION:
097D 1638 :
097D 1639 : THIS ROUTINE MOVES DATA BETWEEN THE PHYSICALLY CONTIGUOUS BUFFER AND
097D 1640 : THE USER'S BUFFER.
097D 1641 :
097D 1642 : INPUTS:
097D 1643 :
097D 1644 : R5 - UCB ADDRESS
097D 1645 :
097D 1646 : OUTPUTS:
097D 1647 :
097D 1648 : DATA MOVE BETWEEN THE PHYSICALLY CONTIGUOUS BUFFER AND THE USER'S BUFFER.
097D 1649 : REGISTER'S R0,R1, AND R2 ARE DESTROYED
097D 1650 :
097D 1651 :--
097D 1652 :
097D 1653 DL_MOVE_TO_BUFFER:
097D 1654 BBS #UCBSV_DL_MAPPING,- ;BUFFER MOVE ROUTINE
097F 1655 UCB$W_DL_FLAGS(R5),10$ ;ADAPTER MAPPING?
0983 1656 CMPB #CDF_READDATA,UCB$B_CEX(R5);READ DATA OPERATION?
0988 1657 BNEQ 10$ ;IF BS YES NOTHING TO MOVE
098A 1658 BBS #0,UCB$B_DL_DCHEK(R5),- ;IF NEQ NOT A READ
098F 1659 10$ ;DATA CHECK IN PROGRESS?
0990 1660 TSTL UCB$A_DL_MOVRTN(R5) ;IF BS YES NOTHING TO MOVE
0994 1661 BEQL 20$ ;ANYTHING TO MOVE?
0996 1662 MOVL UCB$L_DL_BUFADR(R5),R0 ;IF EQL NO
099B 1663 MOVL UCB$A_DL_BUF_VA(R5),R1 ;GET USER BUFFER POINTER
09A0 1664 MOVZWL UCB$W_DL_PBCR(R5),R2 ;GET PHYSICALLY CONTIGUOUS BUFFER ADDRESS
09A5 1665 JSB @UCB$A_DL_MOVRTN(R5) ;GET NUMBER OF BYTES TO TRANSFER
09A9 1666 MOVL R0,UCB$L_DL_BUFADR(R5) ;CALL MOVE ROUTINE
09AE 1667 MOVAB G^IOC$MOVTOUSER,- ;SAVE INTERNAL BUFFER POINTER
09B4 1668 UCB$A_DL_MOVRTN(R5) ;SET NEXT MOVE ROUTINE TO BE USED
09B7 1669 10$: RSB ;RETURN
09B8 1670
09B8 1671 20$: MOVAB G^IOC$MOVTOUSER,- ;SET NEXT MOVE ROUTINE TO BE USED
09BE 1672 UCB$A_DL_MOVRTN(R5) ;
09C1 1673 RSB ;RETURN
09C2 1674
```



```
09C2 1676 .SBTTL MOVE FROM USER BUFFER ROUTINE
09C2 1677 :++
09C2 1678
09C2 1679 DL_MOVE_FROM_BUFFER - MOVE FROM USER BUFFER
09C2 1680
09C2 1681 FUNCTIONAL DESCRIPTION:
09C2 1682
09C2 1683 THIS ROUTINE MOVES DATA BETWEEN THE PHYSICALLY CONTIGUOUS BUFFER AND
09C2 1684 THE USER'S BUFFER.
09C2 1685
09C2 1686 INPUTS:
09C2 1687
09C2 1688 R5 - UCB ADDRESS
09C2 1689
09C2 1690 OUTPUTS:
09C2 1691
09C2 1692 DATA MOVE BETWEEN THE PHYSICALLY CONTIGUOUS BUFFER AND THE USER'S BUFFER.
09C2 1693 REGISTER'S R0,R1, AND R2 ARE DESTROYED
09C2 1694
09C2 1695 :--
09C2 1696
09C2 1697 DL_MOVE_FROM_BUFFER:
09C2 1698 BBS #UCB$V_DL_MAPPING,- :BUFFER MOVE ROUTINE
09C4 1699 UCB$W_DL_FLAGS(R5),10$ :ADAPTER MAPPING?
09C8 1700 CMPB #CDF_WRITEDATA,UCB$B_CEX(R5); :IF BS YES NOTHING TO MOVE
09CD 1701 BNEQ 10$ :WRITE DATA OPERATION?
09CF 1702 BBS #0,UCB$B_DL_DCHEK(R5),- :IF NEQ NOT A WRITE
09D4 1703 10$ :DATA CHECK IN PROGRESS?
09D5 1704 MOVL UCB$L_DL_BUFADR(R5),R0 :IF BS YES NOTHING TO MOVE
09DA 1705 MOVL UCB$A_DL_BUF_VA(R5),R1 :GET USER BUFFER POINTER
09DF 1706 MOVZWL UCB$W_DL_PBCR(R5),R2 :GET PHYSICALLY CONTIGUOUS BUFFER ADDRESS
09E4 1707 JSB @UCB$A_DL_MOVRTN(R5) :GET NUMBER OF BYTES TO TRANSFER
09E8 1708 MOVL R0,UCB$L_DL_BUFADR(R5) :CALL MOVE ROUTINE
09ED 1709 MOVAB G^IOC$MOVFRUSER2,- :SAVE INTERNAL BUFFER POINTER
09F3 1710 UCB$A_DL_MOVRTN(R5) :SET NEXT MOVE ROUTINE TO BE USED
09F6 1711 10$: RSB :RETURN
09F7 1712
09F7 1713 DL_END: :ADDRESS OF LAST LOCATION IN DRIVER
09F7 1714 .END
```

2E 00F6 01 E0  
0093 C5 0B 91  
27 12  
00C9 C5 00 E0  
21 09D4 1703  
50 00DC C5 D0 09D5 1704  
51 00EE C5 D0 09DA 1705  
52 00CC C5 3C 09DF 1706  
00E0 D5 16 09E4 1707  
00DC C5 50 D0 09E8 1708  
00000000 GF 9E 09ED 1709  
00E0 C5 05 09F3 1710  
09F6 1711  
09F7 1712  
09F7 1713  
09F7 1714



DLDRIVER  
Symbol table

- VAX/VMS RL11/RL01,RL02 DISK DRIVER

H 8

16-SEP-1984 00:17:29 VAX/VMS Macro V04-00  
5-SEP-1984 00:12:24 [DRIVER.SRC]DLDRIVER.MAR;1Page 40  
(1)

```
$$$ = 00000020 R 02
$$BASE = 00000001
$$DISPL = 0000000A
$$GENSW = 00000001
$$HIGH = 00000009
$$LIMIT = 00000008
$$LOW = 00000001
$$MNSW = 00000001
$$MXSW = 00000001
$$OP = 00000002
ACPSACCESS ***** X 03
ACPSDEACCESS ***** X 03
ACPSMODIFY ***** X 03
ACPSMOUNT ***** X 03
ACPSREADBLK ***** X 03
ACPSWRITEBLK ***** X 03
ATS_UBA = 00000001
AVAILABLE = 0000036A R 03
BUGS_UNSUPRTCPU ***** X 03
CDF_AVAILABLE = 00000011
CDF_DRVCLR = 00000004
CDF_NOP = 00000010
CDF_OFFSET = 00000006
CDF_PACKACK = 00000008
CDF_READDATA = 0000000C
CDF_READHEAD = 0000000E
CDF_RECAL = 00000003
CDF_RELEASE = 00000005
CDF_RETCENTER = 00000007
CDF_SEARCH = 00000009
CDF_SEEK = 00000002
CDF_UNLOAD = 00000001
CDF_WRITECHECK = 0000000A
CDF_WRITEDATA = 0000000B
CDF_WRITEHEAD = 0000000D
CRBSB_MASK = 0000000E
CRBSL_AUXSTRUC = 00000010
CRBSL_INTD = 00000024
CRBSV_BSY = 00000000
DATA CHECK = 000007C5 R 03
DCS_DISK = 00000001
DDBSK_CART = 00000002
DDBSL_ACPD = 00000010
DDBSL_DDT = 0000000C
DEVSM_AVL = 00040000
DEVSM_DIR = 00000008
DEVSM_ELG = 00400000
DEVSM_FOD = 00004000
DEVSM_IDV = 04000000
DEVSM_NNM = 00000200
DEVSM_ODV = 08000000
DEVSM_RND = 10000000
DEVSM_SHR = 00010000
DL$DDT = 00000000 RG 03
DL_ALIGN = 00000298 R 03
DL_END = 000009F7 R 03
DL_FUNCABLE = 0000005C R 03
```

```
DL_INT = 00000916 RG 03
DL_MOVE_FROM_BUFFER = 000009C2 R 03
DL_MOVE_TO_BUFFER = 0000097D R 03
DL_REGDUMP = 0000095C R 03
DL_RLOX_INIT = 00000143 R 03
DL_RL11_INIT = 000000FC R 03
DL_STARTIO = 000002A8 R 03
DL_UNSLNT = 00000959 R 03
DL_WAIT = 00000263 R 03
DO_FUNCTION = 0000035C R 03
DPTSC_LENGTH = 00000038
DPTSC_VERSION = 00000004
DPT$INITAB = 00000038 R 02
DPTSM_SVP = 00000002
DPT$REINITAB = 00000073 R 02
DPT$TAB = 00000000 R 02
DRCLR = 00000477 R 03
DRVCLR = 0000035C R 03
DTS_RLO1 = 00000009
DTS_RLO2 = 0000000A
DYN$C_CRB = 00000005
DYN$C_DDB = 00000006
DYN$C_DPT = 0000001E
DYN$C_UCB = 00000010
EMBSL_DV_REGSAV = 0000004E
ERL$DEVICERR ***** X 03
ERL$DEVICTMO ***** X 03
EXESABORTIO ***** X 03
EXESALOPHYCNTG ***** X 03
EXESGB_CPUDATA ***** X 03
EXESGB_CPUTYPE ***** X 03
EXESGL_TENUSEC ***** X 03
EXESGL_UBDELAY ***** X 03
EXESIOFORK ***** X 03
EXESLCLDSKVALID ***** X 03
EXESONEPARG ***** X 03
EXESPWRTIMCHK ***** X 03
EXESSENSEMODE ***** X 03
EXESSETCHAR ***** X 03
EXESZEROPARG ***** X 03
FATALERR = 00000398 R 03
FDISPATCH = 00000317 R 03
FEXL = 00000439 R 03
FTAB = 00000038 R 03
FUNCTAB_LEN = 000000A0
FUNCXT = 000003E6 R 03
F_AVAILABLE = 00000000
F_DRVCLR = 00000004
F_GETSTATUS = 00000004
F_NOP = 00000000
F_OFFSET = 00000000
F_PACKACK = 00000004
F_READDATA = 0000000C
F_READHEAD = 00000008
F_RECAL = 00000000
F_RELEASE = 00000000
F_RETCENTER = 00000000
```

DM  
VO



DLDRIVER  
Symbol table

I 8  
- VAX/VMS RL11/RL01,RL02 DISK DRIVER

16-SEP-1984 00:17:29 VAX/VMS Macro V04-00  
5-SEP-1984 00:12:24 [DRIVER.SRC]DLDRIVER.MAR;1

Page 41  
(1)

F_SEARCH	= 00000000		
F_SEEK	= 00000006		
F_UNLOAD	= 00000000		
F_WRITECHECK	= 00000002		
F_WRITEDATA	= 0000000A		
F_WRITEHEAD	= 00000000		
IDBSL_CSR	= 00000000		
IDBSL_OWNER	= 00000004		
IMMED	= 0000047B	R	03
IOSM_DATACHECK	= 00004000		
IOSV_DATACHECK	= 0000000E		
IOSV_INHRETRY	= 0000000F		
IOS_ACCESS	= 00000032		
IOS_ACPCONTROL	= 00000038		
IOS_AVAILABLE	= 00000011		
IOS_CREATE	= 00000033		
IOS_DEACCESS	= 00000034		
IOS_DELETE	= 00000035		
IOS_DRVCLR	= 00000004		
IOS_MODIFY	= 00000036		
IOS_MOUNT	= 00000039		
IOS_NOP	= 00000000		
IOS_PACKACK	= 00000008		
IOS_READHEAD	= 0000000E		
IOS_READBLK	= 00000021		
IOS_READPBLK	= 0000000C		
IOS_READVBLK	= 00000031		
IOS_SEEK	= 00000002		
IOS_SENSECHAR	= 0000001B		
IOS_SENSEMODE	= 00000027		
IOS_SETCHAR	= 0000001A		
IOS_SETMODE	= 00000023		
IOS_UNLOAD	= 00000001		
IOS_VIRTUAL	= 0000003F		
IOS_WRITECHECK	= 0000000A		
IOS_WRITEBLK	= 00000020		
IOS_WRITEPBLK	= 0000000B		
IOS_WRITEVBLK	= 00000030		
IOCSDIAGBUFILL	*****	X	03
IOCSLOADUBAMAP	*****	X	03
IOCSMNTVER	*****	X	03
IOCSMOVFRUSER	*****	X	03
IOCSMOVFRUSER2	*****	X	03
IOCSMOVTOUSER	*****	X	03
IOCSMOVTOUSER2	*****	X	03
IOCSPURGDATAP	*****	X	03
IOCSRELCHAN	*****	X	03
IOCSRELDATAP	*****	X	03
IOCSRELMAPREG	*****	X	03
IOCSREQCOM	*****	X	03
IOCSREQDATAP	*****	X	03
IOCSREQMAPREG	*****	X	03
IOCSREQPCANL	*****	X	03
IOCSRETURN	*****	X	03
IOCSWFIKPCN	*****	X	03
IRPSL_MEDIA	= 00000038		
IRPSL_SVAPTE	= 0000002C		

IRPSS_FCODE	= 00000006		
IRPSV_DIAGBUF	= 00000007		
IRPSV_FCODE	= 00000000		
IRPSV_PHYSIO	= 00000008		
IRPSW_BCNT	= 00000032		
IRPSW_FUNC	= 00000020		
IRPSW_STS	= 0000002A		
MASKH	= 00000008		
MASKL	= 04000000		
MMGSG_L_SPTBASE	*****	X	03
NOP	0000035C	R	03
NORMAL	00000389	R	03
PACKACK	00000362	R	03
POSIT	000004AE	R	03
PR\$_IPL	= 00000012		
PR\$_SID_TYP730	= 00000003		
PR\$_SID_TYP750	= 00000002		
PR\$_SID_TYP780	= 00000001		
PR\$_SID_TYP785	= 00000009		
PR\$_SID_TYP790	= 00000004		
PR\$_SID_TYPMAX	= 00000008		
PR\$_SID_TYPUV1	= 00000007		
PREPROCESS	000002A8	R	03
PTES\$ PFN	= 00000015		
PWRFATL	000008E8	R	03
READDATA	00000379	R	03
READHEAD	00000372	R	03
RESETXFR	000008DB	R	03
RETREG	0000081F	R	03
RETRYERR	0000038F	R	03
RL_BA	00000002		
RL_BAE	00000008		
RL_CS	00000000		
RL_CS_M_CE	= 00008000		
RL_CS_M_CRDY	= 00000080		
RL_CS_M_DRDY	= 00000001		
RL_CS_M_IE	= 00000040		
RL_CS_V_CE	= 0000000F		
RL_CS_V_CRC	= 0000000B		
RL_CS_V_DE	= 0000000E		
RL_CS_V_NXM	= 0000000D		
RL_CS_V_OPI	= 0000000A		
RL_DA	00000004		
RL_DA_M_MRK	= 00000001		
RL_DA_M_RST	= 00000008		
RL_DA_M_STS	= 00000002		
RL_MP	00000006		
RL_MP_M_BH	= 00000008		
RL_MP_M_CHE	= 00004000		
RL_MP_M_DSE	= 00000100		
RL_MP_M_HO	= 00000010		
RL_MP_M_TYP	= 00000080		
RL_MP_M_WDE	= 00008000		
RL_MP_M_WGE	= 00000400		
RL_MP_V_VC	= 00000009		
RL_MP_V_WGE	= 0000000A		
RL_MP_V_WL	= 0000000D		



DLDRIVER  
Symbol table

J 8  
- VAX/VMS RL11/RL01,RL02 DISK DRIVER

16-SEP-1984 00:17:29 VAX/VMS Macro V04-00  
5-SEP-1984 00:12:24 [DRIVER.SRC]DLDRIVER.MAR;1

Page 42  
(1)

RL\_NUM\_REGS = 00000004  
RL\_SLM = 00000005  
SEEK = 0000035C R 03  
SIZ... = 00000001 R 03  
SPECOND = 000008BC  
SS\$CTRLERR = 00000054  
SS\$DATAHECK = 0000005C  
SS\$DRVERR = 0000008C  
SS\$IVBUFLEN = 0000034C  
SS\$MEDOFL = 000001A4  
SS\$NORMAL = 00000001  
SS\$PARITY = 000001F4  
SS\$TIMEOUT = 0000022C  
SS\$VOLINV = 00000254  
SS\$WRITLCK = 0000025C  
UCBSA\_DL\_BUF\_PA = 000000F2  
UCBSA\_DL\_BUF\_VA = 000000EE  
UCBSA\_DL\_MOVRTN = 000000E0  
UCBSB\_CEX = 00000093  
UCBSB\_DEVCLASS = 00000040  
UCBSB\_DEVTYP = 00000041  
UCBSB\_DIPL = 0000005E  
UCBSB\_DL\_DCHEK = 000000C9  
UCBSB\_DL\_DPPE = 000000E4  
UCBSB\_DL\_XBA = 000000EB  
UCBSB\_ERTCNT = 00000080  
UCBSB\_ERTMAX = 00000081  
UCBSB\_FEX = 00000092  
UCBSB\_FIPL = 0000000B  
UCBSB\_SECTORS = 00000044  
UCBSB\_TRACKS = 00000045  
UCBSK\_DL\_BUFSZ = 00000014  
UCBSK\_DL\_LEN = 000000F8  
UCBSK\_LCC\_DISK\_LENGTH = 000000CC  
UCBSL\_CRB = 00000024  
UCBSL\_DEVCHAR = 00000038  
UCBSL\_DEVCHAR2 = 0000003C  
UCBSL\_DL\_BUFADR = 000000DC  
UCBSL\_DL\_DPR = 000000D8  
UCBSL\_DL\_FMPR = 000000DC  
UCBSL\_DL\_PMPR = 000000E0  
UCBSL\_DL\_SVAPTE = 000000D8  
UCBSL\_DPC = 0000009C  
UCBSL\_FPC = 0000000C  
UCBSL\_FR3 = 00000010  
UCBSL\_IRP = 00000058  
UCBSL\_MAXBLOCK = 000000B0  
UCBSL\_MEDIA = 000000BC  
UCBSL\_MEDIA\_ID = 0000008C  
UCBSL\_SVAPTE = 00000078  
UCBSM\_DIAGBUF = 00000002  
UCBSM\_NOCNVRT = 00000004  
UCBSM\_ONLINE = 00000010  
UCBSM\_POWER = 00000020  
UCBSM\_TIMEOUT = 00000040  
UCBSM\_VALID = 00000800  
UCBSV\_DL\_22BIT = 00000000

UCBSV\_DL\_MAPPING = 00000001  
UCBSV\_INT = 00000001  
UCBSV\_POWER = 00000005  
UCBSV\_VALID = 0000000B  
UCBSW\_BCNT = 0000007E  
UCBSW\_BCR = 000000C0  
UCBSW\_BOFF = 0000007C  
UCBSW\_CYLINDERS = 00000046  
UCBSW\_DA = 000000BC  
UCBSW\_DC = 000000BE  
UCBSW\_DEVBUSIZ = 00000042  
UCBSW\_DEVSTS = 00000068  
UCBSW\_DL\_BA = 000000D0  
UCBSW\_DL\_CS = 000000CE  
UCBSW\_DL\_DA = 000000D2  
UCBSW\_DL\_DB = 000000E5  
UCBSW\_DL\_DPN = 000000D6  
UCBSW\_DL\_FLAGS = 000000F6  
UCBSW\_DL\_MP = 000000D4  
UCBSW\_DL\_PBCR = 000000CC  
UCBSW\_DL\_SBA = 000000EC  
UCBSW\_FUNC = 0000009A  
UCBSW\_OFFSET = 000000C8  
UCBSW\_STS = 00000064  
UCBSW\_UNIT = 00000054  
UNLOAD = 0000036A R 03  
UPDATE = 000007DB R 03  
VASM\_BYTE = 000001FF  
VASS\_VPN = 00000015  
VASV\_VPN = 00000009  
VECSB\_DATAPATH = 00000013  
VECSL\_IDB = 00000008  
VECSL\_INITIAL = 0000000C  
VECSL\_UNITINIT = 00000018  
VECSS\_DATAPATH = 00000005  
VECSS\_MAPREG = 0000000F  
VECSV\_DATAPATH = 00000000  
VECSV\_MAPREG = 00000000  
VECSW\_MAPREG = 00000010  
WRITECHECK = 00000372 R 03  
WRITEDATA = 00000379 R 03  
XFER = 0000061A R 03



+-----+  
! Psect synopsis !  
+-----+

PSECT name	Allocation	PSECT No.	Attributes
ABS	00000000 ( 0.)	00 ( 0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
\$AB\$\$	000000FA ( 250.)	01 ( 1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
\$\$\$105_PROLOGUE	00000088 ( 136.)	02 ( 2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
\$\$\$115_DRIVER	000009F7 ( 2551.)	03 ( 3.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG

+-----+  
! Performance indicators !  
+-----+

Phase	Page faults	CPU Time	Elapsed Time
Initialization	30	00:00:00.05	00:00:00.55
Command processing	128	00:00:00.39	00:00:01.95
Pass 1	642	00:00:20.98	00:01:13.05
Symbol table sort	0	00:00:02.56	00:00:09.30
Pass 2	308	00:00:04.50	00:00:21.00
Symbol table output	38	00:00:00.23	00:00:01.81
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	1150	00:00:28.74	00:01:47.68

The working set limit was 2250 pages.  
171426 bytes (335 pages) of virtual memory were used to buffer the intermediate code.  
There were 130 pages of symbol table space allocated to hold 2340 non-local and 91 local symbols.  
1714 source lines were read in Pass 1, producing 24 object records in Pass 2.  
67 pages of virtual memory were used to define 62 macros.

+-----+  
! Macro library statistics !  
+-----+

Macro library name	Macros defined
_\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	40
_\$255\$DUA28:[SYSLIB]STARLET.MLB;2	11
TOTALS (all libraries)	51

2659 GETs were required to define 51 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:DLDRIVER/OBJ=OBJ\$:DLDRIVER MSRC\$:DLDRIVER/UPDATE=(ENH\$:DLDRIVER)+EXECMLS/LIB



0109 AH-BT13A-SE  
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY

